PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-248615

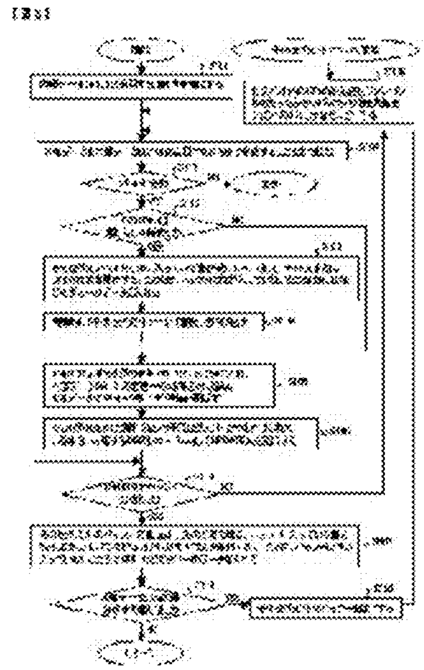
(43)Date of publication of application : 05.09.2003

(51)Int.Cl. G06F 12/00

(21)Application number : 2002-047795 (71)Applicant : NEC CORP

(22)Date of filing : 25.02.2002 (72)Inventor : INAMOTO NAOTA

(54) STORAGE DEVICE OF XML DATA TO RDB, ACQUIRING DEVICE AND METHOD OF XML DATA FROM RDB, AND PROGRAM



(57)Abstract:

PROBLEM TO BE SOLVED: To reduce the volume of buffer necessary to store the XML data having many repeated items in an RDB.

SOLUTION: The XML data is stored in the RDB by tracing the internal data having a structure of the XML data as a storage object, having the correspondence with the RDB, and having the buffer for only one set with respect to the number of repeated parts of the XML data, and simultaneously issuing an SQL sentence performing the data adding operation to the RDB while reading the XML data. Here, with respect to the repeated parts of the XML data, the record is inserted by issuing an INSERT sentence using a dummy value in a column on which the value in the record has not yet determined, to a table storing the value of a tag of the repeated part of the RDB at a time when the value of the tag is accumulated in the buffer by every repeated part (step S107). When the value of the tag corresponding to the value of the column from which the dummy value is written out, is read out from the XML data, an UPDATE

sentence is issued to update the record (step S106).

LEGAL STATUS

[Date of request for examination]	14.01.2005
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

CLAIMS

[Claim(s)]
[Claim 1] Enclosure to RDB of XML data equipped with a storing means store said XML data in said RDB by publishing the memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for storing, and

has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, and the SQL sentence which performs data add operation to said RDB while reading said XML data at the same time it follows said in-house data in order.
[Claim 2] Said storing means about the repeat part of said XML data As opposed to the table which stores the value of the tag of the repeat part of said RDB concerned for every repeat part when the value of a tag is accumulated in a buffer A means by which the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record, Enclosure to RDB of the XML data characterized by including a means to publish a UPDATE sentence and to update said record when the value of the tag corresponding to the value of the column writing out the value of said dummy is read from said XML data.
[Claim 3] The acquisition equipment of the XML data from the RDB equipped with an acquisition means acquire said XML data from said RDB by publishing the memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for acquisition, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, and the SQL sentence which perform data-acquisition actuation to said RDB while writing out said XML data at the same time it follows said in-house data in order.
[Claim 4] Said acquisition means about the repeat part of said XML data A SELECT sentence is published to the table which stores the value of the tag of the repeat part concerned. After accumulating the value of the column in the record concerned in the corresponding buffer for every record, Acquisition equipment of the XML data from RDB including the means which repeats the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data according to claim 1.
[Claim 5] Reading said XML data at the same time it follows in order the in-house data which expresses the structure of the XML data used as the candidate for storing, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB It has the storing step which stores said XML data in said RDB by publishing the SQL sentence which performs data add operation to said RDB. And said storing step About the repeat part of said XML data As opposed to the table which stores the value of the tag of the repeat part of said RDB concerned for every repeat part when the value of a tag is accumulated in a buffer The step in which the value in a record publishes the INSERT sentence which used the dummy

value for the column which has not yet become settled, and inserts a record, The storing approach to RDB of the XML data characterized by including the step which publishes a UPDATE sentence and updates said record when the value of the tag corresponding to the value of the column writing out the value of said dummy is read from said XML data.

[Claim 6] Writing out said XML data at the same time it follows in order the in-house data which expresses the structure of the XML data used as the candidate for acquisition, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB It has the acquisition step which acquires said XML data from said RDB by publishing the SQL sentence which performs data acquisition actuation to said RDB. And said acquisition step About the repeat part of said XML data A SELECT sentence is published to the table which stores the value of the tag of the repeat part concerned. The acquisition approach of the XML data from RDB containing the step which repeats the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data after accumulating the value of the column in the record concerned in the corresponding buffer for every record.

[Claim 7] The computer which constitutes the enclosure to RDB of XML data Reading said XML data at the same time it follows in order the in-house data which expresses the structure of the XML data used as the candidate for storing, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB It is a storing means to store said XML data in said RDB by publishing the SQL sentence which performs data add operation to said RDB. About the repeat part of said XML data As opposed to the table which stores the value of the tag of the repeat part of said RDB concerned for every repeat part when the value of a tag is accumulated in a buffer A means by which the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record, The program operated as a storing means including a means to publish a UPDATE sentence and to update said record when the value of the tag corresponding to the value of the column writing out the value of said dummy is read from said XML data.

[Claim 8] The computer which constitutes the acquisition equipment of the XML data from RDB Writing out said XML data at the same time it follows in order the in-house data which expresses the structure of the XML data used as the candidate for acquisition, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB It is an acquisition means to acquire said XML data from said RDB by publishing the SQL sentence which performs data acquisition actuation to said RDB. About the repeat part of said XML data A SELECT sentence is published to the table which stores the value of the tag of the repeat part concerned. The program operated as an acquisition means including the means which repeats the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data after accumulating the value of the column in the record concerned in the corresponding buffer for every record.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] This invention relates to the equipment which acquires the equipment and XML data which store XML data in RDB from RDB.
[0002]
[Description of the Prior Art] In case the data described by XML (eXtensible Markup Language) are stored in RDB (Relational Data Base), XML data all are not stored by the stream (Stream) as it is, but each column of RDB and the value of each tag of XML are matched, and it stores in some tables so that retrieval etc. can be performed effectively. For this reason, after this conventional kind of enclosure changes XML data into the XML data of other formats which are easy to store in RDB collectively and stores them in a buffer, it is stored in RDB. Moreover, in case the XML data stored in this way are acquired from RDB, XML data were taken out collectively, and it accumulated in the buffer, and has changed into the format of the target XML data

from the format which is easy to store. The tag of XML data of the XML data of the format which is easy to store in RDB is usually XML data of the following formats that the column name of RDB and the value of a tag serve as the contents of the column of RDB.
[0003] <Data> <Record1 <Column1>> </Column1> value1-1 Column <Column2> </Column2> value1-2 Column <Column3> </Column2> 1-3 </Column3> </Record1> <Record2> <Column1> value2-1 </Column1> <Column2> value2-2 Column <Column3> value -- value2-3 </Column3> </Record2> </Data> [0004 --]
[Problem(s) to be Solved by the Invention] Although the DOM (Document Object Model) object which are usually a XSLT (eXtensible Style Language Transformation) processor and a parsing tree is used with the conventional equipment which has such a configuration The memory space which a XSLT processor or a DOM object generally uses as a buffer If the XML data to treat become large, along with it, it will become large. It cannot maintain at the constant value of the same order as the magnitude of the data (DTD (Document Type Definition) or XMLShema) which do not depend the amount of memory which the whole equipment uses on the size of XML data, but express the structure of XML.
[0005] Since the reason is stored in DB after it changes XML data into anhedron-type XML data collectively and stores them in a buffer, when an iteration item performs this about XML data with large a large number size, it is because the amount of use memory is proportional to the size of XML data in almost all cases.
[0006]
[Objects of the Invention] When it stores XML data with many iteration items in RDB, the purpose of this invention does not depend the amount of use memory on the size of XML data, even if a large number [the purpose / XML data / an iteration item], but is to enable it to maintain at the constant value of the same order as the magnitude of the data DTD showing the structure of XML etc.
[0007] When acquiring XML data with many iteration items from RDB, another purpose of this invention does not depend the amount of use memory on the size of XML data, even if a large number [purpose / XML data / an iteration item], but is to enable it to maintain at the constant value of the same order as the magnitude of the data DTD showing the structure of XML etc.
[0008]
[Means for Solving the Problem] The enclosure to RDB of the XML data of this invention The memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for storing, and has a buffer only for

a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, It has a storing means to store said XML data in said RDB, by publishing the SQL sentence which performs data add operation to said RDB, reading said XML data at the same time it follows said in-house data in order. Said storing means more specifically about the repeat part of said XML data As opposed to the table which stores the value of the tag of the repeat part of said RDB concerned for every repeat part when the value of a tag is accumulated in a buffer A means by which the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record, When the value of the tag corresponding to the value of the column writing out the value of said dummy is read from said XML data, a means to publish a UPDATE sentence and to update said record is included.
[0009] Moreover, the acquisition equipment of the XML data from RDB of this invention The memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for acquisition, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, It has an acquisition means to acquire said XML data from said RDB, by publishing the SQL sentence which performs data acquisition actuation to said RDB, writing out said XML data at the same time it follows said in-house data in order. About the repeat part of said XML data, said acquisition means publishes a SELECT sentence to the table which stores the value of the tag of the repeat part concerned, and after it accumulates the value of the column in the record concerned in the corresponding buffer for every record, more specifically, it includes the means which repeats the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data.
[0010]
[Function] For every repeat part, if it is in acquisition equipment from RDB of the XML data of this invention, when the value of a tag is accumulated in a buffer, to the table which stores the value of the tag of the repeat part of RDB concerned, the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record about the repeat part of XML data. And a UPDATE sentence is published and updated when the value of the tag corresponding to the value of the column writing out a dummy value is read from XML data. The table of RDB which stores the value of the tag of the repeat part of XML data by this When it also stores the value of the tag of the outside which is not

included in the repeat part, the beginning of data to the table concerned An INSERT sentence, Although it separates to the multiple times by the UPDATE sentence, however large only the buffer only for a lot may be sufficient about the number of the repeat parts of XML data and the size of XML data may become, the size of an in-house data does not exceed the magnitude of DTD.
[0011] Moreover, if it is in the acquisition equipment of the XML data from RDB of this invention About the repeat part of XML data, a SELECT sentence is published to the table which stores the value of the tag of the repeat part concerned. After accumulating the value of the column in the record concerned in the corresponding buffer for every record, in order to repeat the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data, However large only the buffer only for a lot may be sufficient about the number of the repeat parts of XML data and the size of XML data may become, the size of an in-house data does not exceed the magnitude of DTD.
[0012]
[Embodiment of the Invention] Next, the gestalt of operation of this invention is explained to a detail with reference to a drawing.
[0013] If drawing 1 is referred to, the enclosure 100 to RDB of the XML data in the gestalt of operation of this invention is equipped with the memory 102 which remembers in-house-data DA101 to be the storing means 101 to RDB of XML data, and is connected to the input device 104 which memorizes the XML data 103 used as the candidate for storing, and RDB105 which stores XML data. Input devices 104 are a magnetic disk, a communication buffer, etc. In addition, 110 is computer-readable record media, such as a magnetic disk and semiconductor memory, and the program for storing is recorded. The program for storing recorded here is read by the computer which constitutes enclosure 100, and the storing means 101 as a module is realized on the computer by controlling actuation of the computer.
[0014] In-house-data DA101 is prepared in advance every XML data 103 used as the candidate for an input, expresses the structure of the XML data 103, has correspondence relation with RDB105, and has a buffer only for a lot about the repeat in the XML data 103. In-house-data DA101 has structure of the tree structure which made the repeat part of the XML data 103 the child segment about the data (DTD or XMLShema) showing the structure of the XML data 103, and, more specifically, has the XPath, the table name of RDB105 and column name which are matched with it, and a buffer holding a value about the tag of XML into which a value can go. A parent

segment and a child segment are connected with a pointer.
[0015] Reading the XML data 103 of an input at the same time it follows in-house-data DA101 in order, the storing means 101 operates so that the SQL sentence which performs data add operation to RDB105 may be published. However large the size of the XML data 103 may become at this time, the size of in-house-data DA101 does not exceed the magnitude of DTD.
[0016] Moreover, the acquisition equipment 200 of the XML data from RDB in the gestalt of operation of this invention is equipped with the memory 202 which remembers in-house-data DA201 to be the acquisition means 201 of the XML data from RDB, and is connected to the output unit 204 which memorizes RDB205 which stores XML data, and the acquired XML data 203. Output units 204 are a magnetic disk, a communication buffer, etc. In addition, 210 is computer-readable record media, such as a magnetic disk and semiconductor memory, and the program for acquisition is recorded. The program for acquisition recorded here is read by the computer which constitutes acquisition equipment 200, and the acquisition means 201 as a module is realized on the computer by controlling actuation of the computer.
[0017] In-house-data DA201 is prepared in advance every XML data 203 used as the candidate for acquisition, expresses the structure of the XML data 203, has correspondence relation with RDB205, and has a buffer only for a lot about the repeat in the XML data 203. In-house-data DA201 has structure of the tree structure which made the repeat part of the XML data 203 the child segment about the data (DTD or XMLShema) showing the structure of the XML data 203, and, more specifically, has the XPath, the table name of RDB205 and column name which are matched with it, and a buffer holding a value about the tag of XML into which a value can go. A parent segment and a child segment are connected with a pointer.
[0018] Writing out the XML data 203 at the same time it follows in-house-data DA201 in order, the acquisition means 201 operates so that the SQL sentence which performs data acquisition actuation to RDB205 may be published. However large the size of the XML data 203 may become at this time, the size of in-house-data DA201 does not exceed the magnitude of DTD.
[0019] Next, with reference to the flow chart of drawing 2 , actuation of the whole enclosure 100 of the gestalt of this operation is explained to a detail.
[0020] First, the location to which in-house-data DA101 pays its attention is made into a head (step S101 of drawing 2). Since in-house-data DA101 has the tree structure of a segment, it is set to the location which pays its attention to the head of the segment of the root at first. Next, the XML data 103 are read till the place

corresponding to XPath under view of in-house-data DA101 (step S102). It will be made an error if there is no place which agrees in XPath here in the XML data 103 (it is NO at step S111). Furthermore, in the beginning of a repeat, the tag corresponding to the XPath progresses to step S103 by checking in the beginning of a repeat (step S112), and when it is not the beginning of a repeat, it progresses to step S105. Since the pointer to a child segment is contained in the part of the beginning of a repeat, it becomes clear by the existence of a pointer whether to be the beginning of a repeat.
[0021] At step S103, an INSERT sentence or a UPDATE sentence is published to the table on which RDB105 corresponds based on the value included in a buffer about a parent segment. An INSERT sentence is used at the time of initiation of the first child segment in the parent segment concerned (the beginning of a repeat), and a UPDATE sentence is used at the time of initiation of the child segment of the 2nd henceforth in the parent segment concerned (the beginning of a repeat). In an INSERT sentence, although one line is inserted in a table, if no value of the columns of the line is contained in the buffer, the place in which the value is not contained puts in the data of dummies, such as NULL. Next, the part which begins from ″starting about a child segment″ is recursively performed about the child segment corresponding to it of in-house-data DA101 (step S104). And it progresses to step S113.
[0022] On the other hand, at step S105, the value corresponding to XPath is read from the XML data 103, it stores in the buffer of in-house-data DA101, and the location where a skip and in-house-data DA101 pay their attention to the termination tag of XML corresponding to XPath of the XML data 103 is advanced to a degree. Next, when XPath to which its attention was paid is matched with DB record with which the column of the table of RDB105 published the INSERT sentence from a descendant's segment, a UPDATE sentence is published to those DB records (step S106). And it progresses to step S113.
[0023] At step S113, it confirms whether to be the end of a segment while the view location of an in-house data is paying its attention, and when it is not an end, processing from step S102 is repeated and performed. When there is a value added to the buffer of the segment concerned in the case of an end, based on the value included in a buffer, to RDB105, an INSERT sentence is published in the case of the first beginning, and a UPDATE sentence is published in the case of the beginning to the same record after the 2nd times (step S107). And buffers whether the next part of the XML data 103 is the tag which repeats the segment, and other than the buffer copied at step S109 in order to reuse the buffer of the segment when checking and having become a repeat are emptied (step S108), and processing from step S102 is

repeated and performed. A return is carried out when it is not a repetition. In addition, although it may be made to empty all the buffers of the segment at step S108, the processing from step S109 is repeated in that case.
[0024] thus, in the enclosure 100 of the gestalt of this operation Since the value which it has on memory 102 in-between is constituted so that it may become the magnitude of the structure (DTD) of the XML data 103 when it stores the XML data 103 in RDB105, However large even if and the size of the XML data 103 may become, size of the data which it has on memory 102 in-between is not depended on the size of the input XML data 103, but it is made to the constant value of the same order as the magnitude of DTD. [an iteration item]
[0025] Next, with reference to the flow chart of drawing 3 , actuation of the whole acquisition equipment 200 of the gestalt of this operation is explained to a detail.
[0026] First, the location to which in-house-data DA201 pays its attention is made into a head (step S201 of drawing 3). Since in-house-data DA201 has the tree structure of a segment, it is set to the location which pays its attention to the head of the segment of the root at first. Next, a SELECT sentence is published to the table of RDB205 matched with XPath under view of the segment (step S202). Next, it is confirmed whether there is any next record of the activation result of the SQL sentence published at step S202 (step S203). A return is carried out when there is no following record (it is NO at step S211). When there is the following record, the next record of the activation result of the SQL sentence published at step S202 is fetched, and the acquired value is put into the buffer of the segment of in-house-data DA201 (step S204).
[0027] Next, in the beginning of a repeat, XPath of the view location of in-house-data DA201 confirms whether to be a tag corresponding to the repeat of the XML data 203 (step S212), and it progresses to step S205, and when it is not the beginning of a repeat, it progresses to step S206. Since the pointer to a child segment is contained in the part of the beginning of a repeat, it becomes clear by the existence of a pointer whether to be the beginning of a repeat.
[0028] At step S205, processing from "starting about a child segment" is recursively performed about the child segment of in-house-data DA201 corresponding to the XPath. And it progresses to step S213.
[0029] When the value of the buffer corresponding to XPath under view of in-house-data DA201 is empty, a SELECT sentence is published to RDB205 and the value of a buffer is fill uped with step S206. And the value of the tag corresponding to XPath and a buffer is written out to the XML data 203, and the location under view of

in-house-data DA201 is advanced to a degree (step S207). And it progresses to step S213.
[0030] At step S213, the location under view of in-house-data DA201 confirms whether to be the end of the segment. When it is not the end of a segment, processing from step S212 is repeated and performed. In the case of the end of a segment, processing from step S203 is repeated and performed.
[0031] When the XML data 203 are acquired from RDB205 with the acquisition equipment 200 of the gestalt of this operation, Since the value which writes out the XML data 203 as a stream and it has on memory 202 in-between is constituted so that it may become the magnitude of the structure (DTD) of the XML data 203, However large even if and the size of the XML data 203 may become, size of the data which it has on memory 202 in-between is not depended on the size of the XML data 203, but it is made to the constant value of the same order as the magnitude of DTD.
[an iteration item]
[0032] Next, actuation of the gestalt of this operation is explained using a concrete example.
[0033] Drawing 4 shows the example of DB schema of DTD and RDB of XML data. In DTD, only the element type declaratives is shown and it has the structure where Element Item is repeated. Moreover, RDB has two tables of a PORequest table and a POItem table.
[0034] Drawing 5 shows the example of the correspondence relation between the tag in XML, and the column of DB table. In this example, the value of the tag whose Xpath(s) are "/POReq/PONumber", "/POReq/PODate", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" among the tags of XML into which a value can go is equivalent to the columns POId, Date, and From of a PORequest table, and the value of To. Moreover, Xpath contained in the tag Item of the repeat part of XML The value of the tag which are "/POReq/Item/POItemNumber", "/POReq/Item/ProductId", "/POReq/Item/Request/Amount", and "/POReq/Item/Request/Price", The value of the tag whose Xpath is "/POReq/Location" out of the repeat part concerned is equivalent to the value of the columns POId, ItemNumber, and PId of a POItem table, Quantity, RequestPrice, and Location. In addition, the column POId of a PORequest table and the column POId of a POItem table are in the relation between a primary key and a foreign key.
[0035] Drawing 6 shows the example of the XML data used as the candidate for storing. In this example, the part of iteration items 1-3 has structure repeatedly.

[0036] Drawing 7 shows the example of an in-house data. The in-house data of this example about DTD of drawing 4 showing the structure of the XML data of drawing 6 Have the tree structure which made the repeat part of XML data the child segment (segment 2), and made except [its] the parent segment (segment 1), and it sets to each segments 1 and 2. According to the correspondence relation of drawing 5 , the XPath, the table name of RDB and column name which are matched with it, and the buffer holding a value are set up about the tag of XML into which a value can go. Such an in-house data is created as follows, for example.
[0037] First, a parent segment 1 is generated. Next, if drawing 5 is referred to, since the first Xpath into which the value of XML can go is "/POReq/PONumber" and it corresponds to the column POId of a PORequest table, an entry E1 is generated to a parent segment 1. Similarly, entries E2-E4 are generated. Since the next tag of XML is the tag Item of a repeat, a child segment 2 is generated and the entry E5 which set up the pointer to Xpath and the child segment 2 of Tag Item is generated to a parent segment 1. Next, although it moves to generation of the entry of a child segment 2, since the column POId of a PORequest table and the column POId of a POItem table are in the relation between a primary key and a foreign key when drawing 5 is referred to, after generating the entry E50 with Xpath corresponding to a primary key, the table and column which store a foreign key, and a buffer, the entries E51-E54 for each Xpath into which the value of XML can go are generated. Next, the entry E6 for Xpath of the last at which the value of return and XML can start generation of the entry of a parent segment 1 is generated. Here, in an initial state, the buffer of each entry is empty.
[0038] Hereafter, by DTD and DB schema of drawing 4 , the correspondence relation among these shown in drawing 5 is, the XML data to input show drawing 6 , the case where an in-house data like drawing 7 is prepared is made into an example, and concrete actuation of enclosure 100 is explained.
[0039] The initial state of in-house-data DA101 before beginning to read XML data is what is shown in drawing 7 , and it is set as a location to which the entry E1 of the head of the segment 1 in this pays its attention (step S101). Next, XPath"/POReq/PONumber" corresponding to this location A corresponding place is read from XML data (step S102). Since it is the tag with which a head corresponds, it does not become an error, and since this XPath is not XPath corresponding to the repeat in XML data, it progresses to step S105.
[0040] Value of the tag with which XML data correspond at step S105 "order1" It stores in the buffer of an entry E1 with which an in-house data corresponds, and the

location to which an in-house data pays its attention is advanced to a degree. Since DB column with which XPath to which its attention was paid, and which it was is matched is not the thing of a record which published the INSERT sentence by a descendant's segment, the UPDATE sentence to this is not published (step S106). Next, since it is not the end of a segment, processing from step S102 is repeated and performed. XPath -- "/POReq/PODate", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" ******** -- as a result of repeating the same thing (step S102, step S105), an in-house data is shown in drawing 8 .
[0041] Following XPath"/POReq/Item" If it attaches, since this XPath corresponds to the repeat in XML data, processing of step S103 is performed as processing to the parent segment before starting the processing (processing of a repeat part) based on a child segment 2. Specifically, an INSERT sentence is published to the PORequest table of DB based on the value included in the buffer of the segment 1 of a current in-house data (step S103). Here, the table used as the candidate for actuation is only a table corresponding to that whose buffer of a segment 1 is not empty. For this reason, only a PORequest table is applicable and actuation to the POItem table whose buffer of an entry E6 is empty is not performed.
[0042] Next, processing from step S102 is recursively performed about the child segment 2 corresponding to this XPath"/POReq/Item" (step S104).
[0043] First, the beginning of a child segment 2 XPath"/POReq/PONumber" It corresponds to XPath of an ancestor segment instead of the pass below "/POReq/Item." It is in the relation between a primary key and a foreign key on DB table (this). When [this] becoming clear by whether the same Xpath as Xpath in a child segment is in a parent segment Value which is contained in the buffer at the buffer of parents' segment "order1" is copied (step S109) and it progresses to following XPath"POItemNumber." That is, if the value of the buffer of a parent segment is the primary key of a related ****** sake about the parent table of DB table, and a child table, since there is an item of the foreign key corresponding to a child segment in a child table, a value is copied here.
[0044] next, XPath in a child segment 2 -- "POItemNumber", "ProductId", "Request/Amout", and "Request/Price" ******** -- as a result of performing step S102 and step S105 repeatedly, an in-house data is shown in drawing 9 .
[0045] Since it is the end of a segment 2 now, processing at the time of termination of a segment is performed (step S107). In now, an INSERT sentence is published to the POItem table of DB based on the value included in the buffer of a segment 2. Since the value corresponding to the column Location of a table POItem table at this time is

not yet contained in the buffer of an entry E6, the data of dummies, such as NULL, are used about this column.
[0046] The XML data shown in drawing 6 are XPath of a repeat. "/POReq/Item" Since three are contained, it repeats twice [more], an INSERT sentence is published twice [more] to a POItem table, and the return of the same thing (step S102, step S105, step S107) is carried out to the processing to a segment 1 from the processing about a child segment 2.
[0047] The next XPath of a segment 1 It is "/POReq/Location", and as a result of performing step S102 and step S105 about this, an in-house data is shown in drawing 10 . And since "/POReq/Location" is matched with the column of the record of DB corresponding to a segment 2, a UPDATE sentence is published to the POItem table of DB based on the value with which the INSERT sentence is already contained in this buffer to the record of a POItem table [finishing / issue] (step S106). Next, although activation of step S107 is started, step S107 processes, "only when there are some which were added with the buffer of the segment", and since there is nothing that was newly added to the buffer, in no cases of now, it carries out. By the end of a segment 1, the storing processing to RDB of XML data is ended now. The data stored in RDB at this time are shown in drawing 11 .
[0048] By the above example, the scene of putting in the data of dummies, such as NULL, in step S103, the scene which uses a UPDATE sentence in this step S103, and the scene which uses a UPDATE sentence in step S107 do not appear. These scenes appear, for example by the following examples.
[0049] For example, a case as the entry E2 of drawing 7 is among entries E5 and E6 is considered. In such a case, in step S103 to the parent segment 1 before finishing processing to the entry E4 of a parent segment 1 and starting processing of a repeat part, since the buffer of an entry E2 is still empty, it is put into the data of dummies, such as NULL, by the column Date of a PORequest table. Moreover, if processing of a child segment 1 is finished and it moves to processing of an entry E2, in order that a value may go into the buffer of an entry E2 at step S105, in step S107 at the time of finishing processing of a parent segment 1, the value included in the buffer of an entry E2 uses a UPDATE sentence, and is written in the column Date of a PORequest table.
[0050] Moreover, that a UPDATE sentence is used at step S103 is the case where two or more repeat parts are in a parent segment 1. That is, although an INSERT sentence is used at step S103 to the parent segment at the time of starting the 1st processing of a child segment, a UPDATE sentence is used in the processing of step S103 to the parent segment at the time of starting processing of the child segment

after the 2nd piece.
[0051] Next, by DTD and DB schema of drawing 4 , the correspondence relation among these shown in drawing 5 is, the data stored in RDB show drawing 11 , the case where an in-house data like drawing 7 is prepared is made into an example, and concrete actuation of acquisition equipment 200 is explained.
[0052] The initial state of in-house-data DA201 before beginning the beginning of XML data is what is shown in drawing 7 , and it is set as a location to which the entry E1 of the head of the segment 1 in this pays its attention (step S201). Next, the SELECT sentence which acquires the value of DB column matched by this segment 1 is performed to the PORequest table of DB of the contents shown by drawing 11 (step S202). Since a record exists, it progresses to step S204 (step S203). By fetching the activation result of a SELECT sentence, the value of the buffer of the entries E1-E4 of a segment 1 is buried (step S204). At this time, in-house-data DA201 is shown in drawing 12 .
[0053] Since the first XPath"POReq/PONumber" is not the item of a repeat, it progresses to step S206. Since the corresponding buffer of an entry E1 is not empty, a SELECT sentence is not published (step S206). The tag corresponding to XPath"POReq/PONumber" of in-house-data DA201 and its value are written out to XML data, and progress to the next XPath (step S207). Here, about the termination tag of XML when writing out to XML data, the beginning is delayed until the place surrounded by the tag is closed.
[0054] next, XPath -- "/POReq/POData", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" ******** -- XPath corresponding to [as a result of repeating the same thing (step S207)] the repeat of XML data -- "/POReq/Item" The XML data of a before are written out.
[0055] Since following XPath"/POReq/Item" corresponds to the repeat in XML data, processing from "starting about a child segment" is recursively performed about the child segment 2 corresponding to XPath "/POReq/Item" (step S205). XPath"/POReq/PONumber" of a child segment 2 is a value which corresponds to XPath of an ancestor segment instead of the pass below "/POReq/Item", has become what is in the relation between a primary key and a foreign key on DB table, and is contained in the buffer of an entry E50 in this case at the buffer of the entry E1 of a parent segment. "order1" is copied and it progresses to XPath"POItemNumber" (step S208). Next, the SELECT sentence which acquires the value of DB column matched by this segment 2 is performed to the POItem table of DB of the contents shown by drawing 11 (step S202). Since a record exists, it progresses to step S204

(step S203). By fetching the activation result of a SELECT sentence, the value of the buffer of the entries E51-E54 of a segment 2 is buried (step S204).
[0056] next, XPath -- "POItemNumber", "ProductId", "Request/Amout", and "Request/Price" ******** -- as a result of performing step S207 repeatedly, the 1st of the iteration items corresponding to XPath"POReq/Item" is written out to XML data. Since it is the end of a segment 2 now, the processing from step S203 is repeated. Since the POItem table of DB shown in drawing 11 contains three records, it repeats the same thing (step S203, step S204, step S207) twice [more], writes out the contents corresponding to the two remaining iteration items to XML data, and they carry out a return to the processing to a segment 1 from the processing about a child segment 2.
[0057] The next XPath of a segment 1 It is "/POReq/Location", and since the buffer of the entry E6 corresponding to this is empty, a SELECT sentence is published and fetched to the POItem table of DB, a buffer is buried (step S206), and the contents which perform step S207 and correspond are written out to XML data. Now, the acquisition processing from RDB of XML data is ended. At this time, in-house-data DA201 becomes what is shown in drawing 13 , and the written-out XML data become the same thing as what is shown in drawing 6 .
[0058] Although the gestalt of operation of this invention was explained above, this invention is not limited to the above example, but, in addition to this, various kinds of addition modification is possible for it. For example, when not performing storing and acquisition to coincidence, you may make it prepare one in-house data in common with enclosure 101 and acquisition equipment 201 in drawing 1 , since an in-house data becomes the same with enclosure 101 and acquisition equipment 201 about the XML data of the same structure although in-house-data DA101 and in-house-data DA201 were formed in each of enclosure 101 and acquisition equipment 201. In this case, in accordance with enclosure 101 and acquisition equipment 201, it may be made to realize as one equipment as storing / acquisition equipment.
[0059]
[Effect of the Invention] There is effectiveness which is suppressed by the magnitude to which the amount of use memory is proportional to the size of DTD however large the size of XML data might become when according to this invention XML data were stored in RDB and XML data were acquired from RDB as explained above, is not based on the size of XML data but is suppressed by the constant value of the same order as the magnitude of DTD. The reason has on memory only the in-house data which is proportional to the magnitude of DTD in this invention to needing the memory space

which is proportional to the size of XML data in the existing XML processor, but if other data are storing in DB, they will be written out to DB, and it is for writing out to the stream which creates XML data at the time of the ejection from DB.

[Translation done.]

TECHNICAL FIELD

[Field of the Invention] This invention relates to the equipment which acquires the equipment and XML data which store XML data in RDB from RDB.

[Translation done.]

PRIOR ART

[Description of the Prior Art] In case the data described by XML (eXtensible Markup Language) are stored in RDB (Relational Data Base), XML data all are not stored by the stream (Stream) as it is, but each column of RDB and the value of each tag of XML are matched, and it stores in some tables so that retrieval etc. can be performed effectively. For this reason, after this conventional kind of enclosure changes XML data into the XML data of other formats which are easy to store in RDB collectively and stores them in a buffer, it is stored in RDB. Moreover, in case the XML data stored in this way are acquired from RDB, XML data were taken out collectively, and it accumulated in the buffer, and has changed into the format of the target XML data from the format which is easy to store. The tag of XML data of the XML data of the format which is easy to store in RDB is usually XML data of the following formats that the column name of RDB and the value of a tag serve as the contents of the column of RDB.
[0003] <Data><Record1><Column1> value1-1 </Column1><Column2> value1-2 </Column2><Column3> value1-3 </Column3></Record1><Record2><Column1> value2-1 </Column1><Column2> value2-2 </Column2><Column3> value2-3 </Column3></Record2></Data>

[Translation done.]

EFFECT OF THE INVENTION

[Effect of the Invention] There is effectiveness which is suppressed by the magnitude to which the amount of use memory is proportional to the size of DTD however large the size of XML data might become when according to this invention XML data were stored in RDB and XML data were acquired from RDB as explained above, is not based on the size of XML data but is suppressed by the constant value of the same order as the magnitude of DTD. The reason has on memory only the in-house data which is proportional to the magnitude of DTD in this invention to needing the memory space which is proportional to the size of XML data in the existing XML processor, but if other data are storing in DB, they will be written out to DB, and it is for writing out to the stream which creates XML data at the time of the ejection from DB.

[Translation done.]

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] Although the DOM (Document Object Model) object which are usually a XSLT (eXtensible Style Language Transformation) processor and a parsing tree is used with the conventional equipment which has such a configuration The memory space which a XSLT processor or a DOM object generally uses as a buffer If the XML data to treat become large, along with it, it will become large. It cannot maintain at the constant value of the same order as the magnitude of the data (DTD (Document Type Definition) or XMLShema) which do not depend the amount of memory which the whole equipment uses on the size of XML data, but express the structure of XML.
[0005] Since the reason is stored in DB after it changes XML data into anhedron-type XML data collectively and stores them in a buffer, when an iteration item performs this about XML data with large a large number size, it is because the amount of use memory is proportional to the size of XML data in almost all cases.
[0006]

[Objects of the Invention] When it stores XML data with many iteration items in RDB, the purpose of this invention does not depend the amount of use memory on the size of XML data, even if a large number [the purpose / XML data / an iteration item], but is to enable it to maintain at the constant value of the same order as the magnitude of the data DTD showing the structure of XML etc.
[0007] When acquiring XML data with many iteration items from RDB, another purpose of this invention does not depend the amount of use memory on the size of XML data, even if a large number [purpose / XML data / an iteration item], but is to enable it to maintain at the constant value of the same order as the magnitude of the data DTD showing the structure of XML etc.

[Translation done.]

MEANS

[Means for Solving the Problem] The enclosure to RDB of the XML data of this invention The memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for storing, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, It has a storing means to store said XML data in said RDB, by publishing the SQL sentence which performs data add operation to said RDB, reading said XML data at the same time it follows said in-house data in order. Said storing means more specifically about the repeat part of said XML data As opposed to the table which stores the value of the tag of the repeat part of said RDB concerned for every repeat part when the value of a tag is accumulated in a buffer A means by which the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record, When the value of the tag corresponding to the value of the column writing out the value of said dummy is read from said XML data, a means to publish a UPDATE sentence and to update said record is included.

[0009] Moreover, the acquisition equipment of the XML data from RDB of this invention The memory which memorizes the in-house data which expresses the structure of the XML data used as the candidate for acquisition, and has a buffer only for a lot about the number of the repeat parts of said XML data with correspondence relation with RDB, It has an acquisition means to acquire said XML data from said RDB, by publishing the SQL sentence which performs data acquisition actuation to said RDB, writing out said XML data at the same time it follows said in-house data in order. About the repeat part of said XML data, said acquisition means publishes a SELECT sentence to the table which stores the value of the tag of the repeat part concerned, and after it accumulates the value of the column in the record concerned in the corresponding buffer for every record, more specifically, it includes the means which repeats the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data.

[Translation done.]

OPERATION

[Function] For every repeat part, if it is in acquisition equipment from RDB of the XML data of this invention, when the value of a tag is accumulated in a buffer, to the table which stores the value of the tag of the repeat part of RDB concerned, the value in a record publishes the INSERT sentence which used the dummy value for the column which has not yet become settled, and inserts a record about the repeat part of XML data. And a UPDATE sentence is published and updated when the value of the tag corresponding to the value of the column writing out a dummy value is read from XML data. The table of RDB which stores the value of the tag of the repeat part of XML data by this When it also stores the value of the tag of the outside which is not included in the repeat part, the beginning of data to the table concerned An INSERT sentence, Although it separates to the multiple times by the UPDATE sentence, however large only the buffer only for a lot may be sufficient about the number of the repeat parts of XML data and the size of XML data may become, the size of an in-house data does not exceed the magnitude of DTD.
[0011] Moreover, if it is in the acquisition equipment of the XML data from RDB of this invention About the repeat part of XML data, a SELECT sentence is published to the table which stores the value of the tag of the repeat part concerned. After accumulating the value of the column in the record concerned in the corresponding buffer for every record, in order to repeat the processing which reads the value corresponding to the tag of the repeat part concerned from said buffer, and is written out to the stream of said XML data, However large only the buffer only for a lot may be sufficient about the number of the repeat parts of XML data and the size of XML data may become, the size of an in-house data does not exceed the magnitude of DTD.
[0012]
[Embodiment of the Invention] Next, the gestalt of operation of this invention is explained to a detail with reference to a drawing.
[0013] If drawing 1 is referred to, the enclosure 100 to RDB of the XML data in the gestalt of operation of this invention is equipped with the memory 102 which remembers in-house-data DA101 to be the storing means 101 to RDB of XML data, and is connected to the input device 104 which memorizes the XML data 103 used as the candidate for storing, and RDB105 which stores XML data. Input devices 104 are a magnetic disk, a communication buffer, etc. In addition, 110 is computer-readable record media, such as a magnetic disk and semiconductor memory, and the program for storing is recorded. The program for storing recorded here is read by the computer which constitutes enclosure 100, and the storing means 101 as a module is realized on

the computer by controlling actuation of the computer.
[0014] In-house-data DA101 is prepared in advance every XML data 103 used as the candidate for an input, expresses the structure of the XML data 103, has correspondence relation with RDB105, and has a buffer only for a lot about the repeat in the XML data 103. In-house-data DA101 has structure of the tree structure which made the repeat part of the XML data 103 the child segment about the data (DTD or XMLShema) showing the structure of the XML data 103, and, more specifically, has the XPath, the table name of RDB105 and column name which are matched with it, and a buffer holding a value about the tag of XML into which a value can go. A parent segment and a child segment are connected with a pointer.
[0015] Reading the XML data 103 of an input at the same time it follows in-house-data DA101 in order, the storing means 101 operates so that the SQL sentence which performs data add operation to RDB105 may be published. However large the size of the XML data 103 may become at this time, the size of in-house-data DA101 does not exceed the magnitude of DTD.
[0016] Moreover, the acquisition equipment 200 of the XML data from RDB in the gestalt of operation of this invention is equipped with the memory 202 which remembers in-house-data DA201 to be the acquisition means 201 of the XML data from RDB, and is connected to the output unit 204 which memorizes RDB205 which stores XML data, and the acquired XML data 203. Output units 204 are a magnetic disk, a communication buffer, etc. In addition, 210 is computer-readable record media, such as a magnetic disk and semiconductor memory, and the program for acquisition is recorded. The program for acquisition recorded here is read by the computer which constitutes acquisition equipment 200, and the acquisition means 201 as a module is realized on the computer by controlling actuation of the computer.
[0017] In-house-data DA201 is prepared in advance every XML data 203 used as the candidate for acquisition, expresses the structure of the XML data 203, has correspondence relation with RDB205, and has a buffer only for a lot about the repeat in the XML data 203. In-house-data DA201 has structure of the tree structure which made the repeat part of the XML data 203 the child segment about the data (DTD or XMLShema) showing the structure of the XML data 203, and, more specifically, has the XPath, the table name of RDB205 and column name which are matched with it, and a buffer holding a value about the tag of XML into which a value can go. A parent segment and a child segment are connected with a pointer.
[0018] Writing out the XML data 203 at the same time it follows in-house-data DA201 in order, the acquisition means 201 operates so that the SQL sentence which

performs data acquisition actuation to RDB205 may be published. However large the size of the XML data 203 may become at this time, the size of in-house-data DA201 does not exceed the magnitude of DTD.
[0019] Next, with reference to the flow chart of drawing 2 , actuation of the whole enclosure 100 of the gestalt of this operation is explained to a detail.
[0020] First, the location to which in-house-data DA101 pays its attention is made into a head (step S101 of drawing 2). Since in-house-data DA101 has the tree structure of a segment, it is set to the location which pays its attention to the head of the segment of the root at first. Next, the XML data 103 are read till the place corresponding to XPath under view of in-house-data DA101 (step S102). It will be made an error if there is no place which agrees in XPath here in the XML data 103 (it is NO at step S111). Furthermore, in the beginning of a repeat, the tag corresponding to the XPath progresses to step S103 by checking in the beginning of a repeat (step S112), and when it is not the beginning of a repeat, it progresses to step S105. Since the pointer to a child segment is contained in the part of the beginning of a repeat, it becomes clear by the existence of a pointer whether to be the beginning of a repeat.
[0021] At step S103, an INSERT sentence or a UPDATE sentence is published to the table on which RDB105 corresponds based on the value included in a buffer about a parent segment. An INSERT sentence is used at the time of initiation of the first child segment in the parent segment concerned (the beginning of a repeat), and a UPDATE sentence is used at the time of initiation of the child segment of the 2nd henceforth in the parent segment concerned (the beginning of a repeat). In an INSERT sentence, although one line is inserted in a table, if no value of the columns of the line is contained in the buffer, the place in which the value is not contained puts in the data of dummies, such as NULL. Next, the part which begins from "starting about a child segment" is recursively performed about the child segment corresponding to it of in-house-data DA101 (step S104). And it progresses to step S113.
[0022] On the other hand, at step S105, the value corresponding to XPath is read from the XML data 103, it stores in the buffer of in-house-data DA101, and the location where a skip and in-house-data DA101 pay their attention to the termination tag of XML corresponding to XPath of the XML data 103 is advanced to a degree. Next, when XPath to which its attention was paid is matched with DB record with which the column of the table of RDB105 published the INSERT sentence from a descendant's segment, a UPDATE sentence is published to those DB records (step S106). And it progresses to step S113.
[0023] At step S113, it confirms whether to be the end of a segment while the view

location of an in-house data is paying its attention, and when it is not an end, processing from step S102 is repeated and performed. When there is a value added to the buffer of the segment concerned in the case of an end, based on the value included in a buffer, to RDB105, an INSERT sentence is published in the case of the first beginning, and a UPDATE sentence is published in the case of the beginning to the same record after the 2nd times (step S107). And buffers whether the next part of the XML data 103 is the tag which repeats the segment, and other than the buffer copied at step S109 in order to reuse the buffer of the segment when checking and having become a repeat are emptied (step S108), and processing from step S102 is repeated and performed. A return is carried out when it is not a repetition. In addition, although it may be made to empty all the buffers of the segment at step S108, the processing from step S109 is repeated in that case.
[0024] thus, in the enclosure 100 of the gestalt of this operation Since the value which it has on memory 102 in-between is constituted so that it may become the magnitude of the structure (DTD) of the XML data 103 when it stores the XML data 103 in RDB105, However large even if and the size of the XML data 103 may become, size of the data which it has on memory 102 in-between is not depended on the size of the input XML data 103, but it is made to the constant value of the same order as the magnitude of DTD. [an iteration item]
[0025] Next, with reference to the flow chart of drawing 3 , actuation of the whole acquisition equipment 200 of the gestalt of this operation is explained to a detail.
[0026] First, the location to which in-house-data DA201 pays its attention is made into a head (step S201 of drawing 3). Since in-house-data DA201 has the tree structure of a segment, it is set to the location which pays its attention to the head of the segment of the root at first. Next, a SELECT sentence is published to the table of RDB205 matched with XPath under view of the segment (step S202). Next, it is confirmed whether there is any next record of the activation result of the SQL sentence published at step S202 (step S203). A return is carried out when there is no following record (it is NO at step S211). When there is the following record, the next record of the activation result of the SQL sentence published at step S202 is fetched, and the acquired value is put into the buffer of the segment of in-house-data DA201 (step S204).
[0027] Next, in the beginning of a repeat, XPath of the view location of in-house-data DA201 confirms whether to be a tag corresponding to the repeat of the XML data 203 (step S212), and it progresses to step S205, and when it is not the beginning of a repeat, it progresses to step S206. Since the pointer to a child segment is contained

in the part of the beginning of a repeat, it becomes clear by the existence of a pointer whether to be the beginning of a repeat.
[0028] At step S205, processing from "starting about a child segment" is recursively performed about the child segment of in-house-data DA201 corresponding to the XPath. And it progresses to step S213.
[0029] When the value of the buffer corresponding to XPath under view of in-house-data DA201 is empty, a SELECT sentence is published to RDB205 and the value of a buffer is fill uped with step S206. And the value of the tag corresponding to XPath and a buffer is written out to the XML data 203, and the location under view of in-house-data DA201 is advanced to a degree (step S207). And it progresses to step S213.
[0030] At step S213, the location under view of in-house-data DA201 confirms whether to be the end of the segment. When it is not the end of a segment, processing from step S212 is repeated and performed. In the case of the end of a segment, processing from step S203 is repeated and performed.
[0031] When the XML data 203 are acquired from RDB205 with the acquisition equipment 200 of the gestalt of this operation, Since the value which writes out the XML data 203 as a stream and it has on memory 202 in-between is constituted so that it may become the magnitude of the structure (DTD) of the XML data 203, However large even if and the size of the XML data 203 may become, size of the data which it has on memory 202 in-between is not depended on the size of the XML data 203, but it is made to the constant value of the same order as the magnitude of DTD.
[an iteration item]
[0032] Next, actuation of the gestalt of this operation is explained using a concrete example.
[0033] Drawing 4 shows the example of DB schema of DTD and RDB of XML data. In DTD, only the element type declaratives is shown and it has the structure where Element Item is repeated. Moreover, RDB has two tables of a PORequest table and a POItem table.
[0034] Drawing 5 shows the example of the correspondence relation between the tag in XML, and the column of DB table. In this example, the value of the tag whose Xpath(s) are "/POReq/PONumber", "/POReq/PODate", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" among the tags of XML into which a value can go is equivalent to the columns POId, Date, and From of a PORequest table, and the value of To. Moreover, Xpath contained in the tag Item of the repeat part of XML The value of the tag which are

″/POReq/Item/POItemNumber″, ″/POReq/Item/ProductId″, ″/POReq/Item/Request/Amount″, and ″/POReq/Item/Request/Price″, The value of the tag whose Xpath is ″/POReq/Location″ out of the repeat part concerned is equivalent to the value of the columns POId, ItemNumber, and PId of a POItem table, Quantity, RequestPrice, and Location. In addition, the column POId of a PORequest table and the column POId of a POItem table are in the relation between a primary key and a foreign key.

[0035] Drawing 6 shows the example of the XML data used as the candidate for storing. In this example, the part of iteration items 1-3 has structure repeatedly.

[0036] Drawing 7 shows the example of an in-house data. The in-house data of this example about DTD of drawing 4 showing the structure of the XML data of drawing 6 Have the tree structure which made the repeat part of XML data the child segment (segment 2), and made except [its] the parent segment (segment 1), and it sets to each segments 1 and 2. According to the correspondence relation of drawing 5 , the XPath, the table name of RDB and column name which are matched with it, and the buffer holding a value are set up about the tag of XML into which a value can go. Such an in-house data is created as follows, for example.

[0037] First, a parent segment 1 is generated. Next, if drawing 5 is referred to, since the first Xpath into which the value of XML can go is ″/POReq/PONumber″ and it corresponds to the column POId of a PORequest table, an entry E1 is generated to a parent segment 1. Similarly, entries E2-E4 are generated. Since the next tag of XML is the tag Item of a repeat, a child segment 2 is generated and the entry E5 which set up the pointer to Xpath and the child segment 2 of Tag Item is generated to a parent segment 1. Next, although it moves to generation of the entry of a child segment 2, since the column POId of a PORequest table and the column POId of a POItem table are in the relation between a primary key and a foreign key when drawing 5 is referred to, after generating the entry E50 with Xpath corresponding to a primary key, the table and column which store a foreign key, and a buffer, the entries E51-E54 for each Xpath into which the value of XML can go are generated. Next, the entry E6 for Xpath of the last at which the value of return and XML can start generation of the entry of a parent segment 1 is generated. Here, in an initial state, the buffer of each entry is empty.

[0038] Hereafter, by DTD and DB schema of drawing 4 , the correspondence relation among these shown in drawing 5 is, the XML data to input show drawing 6 , the case where an in-house data like drawing 7 is prepared is made into an example, and concrete actuation of enclosure 100 is explained.

[0039] The initial state of in-house-data DA101 before beginning to read XML data is what is shown in drawing 7 , and it is set as a location to which the entry E1 of the head of the segment 1 in this pays its attention (step S101). Next, XPath"/POReq/PONumber" corresponding to this location A corresponding place is read from XML data (step S102). Since it is the tag with which a head corresponds, it does not become an error, and since this XPath is not XPath corresponding to the repeat in XML data, it progresses to step S105.
[0040] Value of the tag with which XML data correspond at step S105 "order1" It stores in the buffer of an entry E1 with which an in-house data corresponds, and the location to which an in-house data pays its attention is advanced to a degree. Since DB column with which XPath to which its attention was paid, and which it was is matched is not the thing of a record which published the INSERT sentence by a descendant's segment, the UPDATE sentence to this is not published (step S106). Next, since it is not the end of a segment, processing from step S102 is repeated and performed. XPath -- "/POReq/PODate", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" ******** -- as a result of repeating the same thing (step S102, step S105), an in-house data is shown in drawing 8 .
[0041] Following XPath"/POReq/Item" If it attaches, since this XPath corresponds to the repeat in XML data, processing of step S103 is performed as processing to the parent segment before starting the processing (processing of a repeat part) based on a child segment 2. Specifically, an INSERT sentence is published to the PORequest table of DB based on the value included in the buffer of the segment 1 of a current in-house data (step S103). Here, the table used as the candidate for actuation is only a table corresponding to that whose buffer of a segment 1 is not empty. For this reason, only a PORequest table is applicable and actuation to the POItem table whose buffer of an entry E6 is empty is not performed.
[0042] Next, processing from step S102 is recursively performed about the child segment 2 corresponding to this XPath"/POReq/Item" (step S104).
[0043] First, the beginning of a child segment 2 XPath"/POReq/PONumber" It corresponds to XPath of an ancestor segment instead of the pass below "/POReq/Item." It is in the relation between a primary key and a foreign key on DB table (this). When [this] becoming clear by whether the same Xpath as Xpath in a child segment is in a parent segment Value which is contained in the buffer at the buffer of parents' segment "order1" is copied (step S109) and it progresses to following XPath"POItemNumber." That is, if the value of the buffer of a parent segment is the primary key of a related ****** sake about the parent table of DB

table, and a child table, since there is an item of the foreign key corresponding to a child segment in a child table, a value is copied here.
[0044] next, XPath in a child segment 2 -- "POItemNumber", "ProductId", "Request/Amout", and "Request/Price" ******** -- as a result of performing step S102 and step S105 repeatedly, an in-house data is shown in drawing 9 .
[0045] Since it is the end of a segment 2 now, processing at the time of termination of a segment is performed (step S107). In now, an INSERT sentence is published to the POItem table of DB based on the value included in the buffer of a segment 2. Since the value corresponding to the column Location of a table POItem table at this time is not yet contained in the buffer of an entry E6, the data of dummies, such as NULL, are used about this column.
[0046] The XML data shown in drawing 6 are XPath of a repeat. "/POReq/Item" Since three are contained, it repeats twice [more], an INSERT sentence is published twice [more] to a POItem table, and the return of the same thing (step S102, step S105, step S107) is carried out to the processing to a segment 1 from the processing about a child segment 2.
[0047] The next XPath of a segment 1 It is "/POReq/Location", and as a result of performing step S102 and step S105 about this, an in-house data is shown in drawing 10 . And since "/POReq/Location" is matched with the column of the record of DB corresponding to a segment 2, a UPDATE sentence is published to the POItem table of DB based on the value with which the INSERT sentence is already contained in this buffer to the record of a POItem table [finishing / issue] (step S106). Next, although activation of step S107 is started, step S107 processes, "only when there are some which were added with the buffer of the segment", and since there is nothing that was newly added to the buffer, in no cases of now, it carries out. By the end of a segment 1, the storing processing to RDB of XML data is ended now. The data stored in RDB at this time are shown in drawing 11 .
[0048] By the above example, the scene of putting in the data of dummies, such as NULL, in step S103, the scene which uses a UPDATE sentence in this step S103, and the scene which uses a UPDATE sentence in step S107 do not appear. These scenes appear, for example by the following examples.
[0049] For example, a case as the entry E2 of drawing 7 is among entries E5 and E6 is considered. In such a case, in step S103 to the parent segment 1 before finishing processing to the entry E4 of a parent segment 1 and starting processing of a repeat part, since the buffer of an entry E2 is still empty, it is put into the data of dummies, such as NULL, by the column Date of a PORequest table. Moreover, if processing of a

child segment 1 is finished and it moves to processing of an entry E2, in order that a value may go into the buffer of an entry E2 at step S105, in step S107 at the time of finishing processing of a parent segment 1, the value included in the buffer of an entry E2 uses a UPDATE sentence, and is written in the column Date of a PORequest table.
[0050] Moreover, that a UPDATE sentence is used at step S103 is the case where two or more repeat parts are in a parent segment 1. That is, although an INSERT sentence is used at step S103 to the parent segment at the time of starting the 1st processing of a child segment, a UPDATE sentence is used in the processing of step S103 to the parent segment at the time of starting processing of the child segment after the 2nd piece.
[0051] Next, by DTD and DB schema of drawing 4 , the correspondence relation among these shown in drawing 5 is, the data stored in RDB show drawing 11 , the case where an in-house data like drawing 7 is prepared is made into an example, and concrete actuation of acquisition equipment 200 is explained.
[0052] The initial state of in-house-data DA201 before beginning the beginning of XML data is what is shown in drawing 7 , and it is set as a location to which the entry E1 of the head of the segment 1 in this pays its attention (step S201). Next, the SELECT sentence which acquires the value of DB column matched by this segment 1 is performed to the PORequest table of DB of the contents shown by drawing 11 (step S202). Since a record exists, it progresses to step S204 (step S203). By fetching the activation result of a SELECT sentence, the value of the buffer of the entries E1-E4 of a segment 1 is buried (step S204). At this time, in-house-data DA201 is shown in drawing 12 .
[0053] Since the first XPath"POReq/PONumber" is not the item of a repeat, it progresses to step S206. Since the corresponding buffer of an entry E1 is not empty, a SELECT sentence is not published (step S206). The tag corresponding to XPath"POReq/PONumber" of in-house-data DA201 and its value are written out to XML data, and progress to the next XPath (step S207). Here, about the termination tag of XML when writing out to XML data, the beginning is delayed until the place surrounded by the tag is closed.
[0054] next, XPath -- "/POReq/POData", "/POReq/fromRole/BusinessId", and "/POReq/toRole/BusinessId" ******** -- XPath corresponding to [as a result of repeating the same thing (step S207)] the repeat of XML data -- "/POReq/Item" The XML data of a before are written out.
[0055] Since following XPath"/POReq/Item" corresponds to the repeat in XML data, processing from "starting about a child segment" is recursively performed about the

child segment 2 corresponding to XPath "/POReq/Item" (step S205). XPath"/POReq/PONumber" of a child segment 2 is a value which corresponds to XPath of an ancestor segment instead of the pass below "/POReq/Item", has become what is in the relation between a primary key and a foreign key on DB table, and is contained in the buffer of an entry E50 in this case at the buffer of the entry E1 of a parent segment. "order1" is copied and it progresses to XPath"POItemNumber" (step S208). Next, the SELECT sentence which acquires the value of DB column matched by this segment 2 is performed to the POItem table of DB of the contents shown by drawing 11 (step S202). Since a record exists, it progresses to step S204 (step S203). By fetching the activation result of a SELECT sentence, the value of the buffer of the entries E51-E54 of a segment 2 is buried (step S204).
[0056] next, XPath -- "POItemNumber", "ProductId", "Request/Amout", and "Request/Price" ******** -- as a result of performing step S207 repeatedly, the 1st of the iteration items corresponding to XPath"POReq/Item" is written out to XML data. Since it is the end of a segment 2 now, the processing from step S203 is repeated. Since the POItem table of DB shown in drawing 11 contains three records, it repeats the same thing (step S203, step S204, step S207) twice [more], writes out the contents corresponding to the two remaining iteration items to XML data, and they carry out a return to the processing to a segment 1 from the processing about a child segment 2.
[0057] The next XPath of a segment 1 It is "/POReq/Location", and since the buffer of the entry E6 corresponding to this is empty, a SELECT sentence is published and fetched to the POItem table of DB, a buffer is buried (step S206), and the contents which perform step S207 and correspond are written out to XML data. Now, the acquisition processing from RDB of XML data is ended. At this time, in-house-data DA201 becomes what is shown in drawing 13 , and the written-out XML data become the same thing as what is shown in drawing 6 .
[0058] Although the gestalt of operation of this invention was explained above, this invention is not limited to the above example, but, in addition to this, various kinds of addition modification is possible for it. For example, when not performing storing and acquisition to coincidence, you may make it prepare one in-house data in common with enclosure 101 and acquisition equipment 201 in drawing 1 , since an in-house data becomes the same with enclosure 101 and acquisition equipment 201 about the XML data of the same structure although in-house-data DA101 and in-house-data DA201 were formed in each of enclosure 101 and acquisition equipment 201. In this case, in accordance with enclosure 101 and acquisition equipment 201, it may be

made to realize as one equipment as storing / acquisition equipment.

[Translation done.]

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is the block diagram showing the configuration of the gestalt of operation of this invention.
[Drawing 2] It is the flow chart showing actuation of the storing means to RDB of the XML data of the gestalt of operation of this invention.
[Drawing 3] It is the flow chart showing actuation of the acquisition means of the XML data from RDB of the gestalt of operation of this invention.
[Drawing 4] It is drawing showing the example of DTD and DB schema information.
[Drawing 5] It is drawing showing the example of the correspondence relation between the tag in XML, and the column of DB table.
[Drawing 6] It is drawing showing the example of XML data.
[Drawing 7] It is drawing showing the example of the in-house data of an initial state.
[Drawing 8] It is drawing showing the condition of the in-house data of the midst which stores XML data in RDB.
[Drawing 9] It is drawing showing the condition of the in-house data of the midst which stores XML data in RDB.
[Drawing 10] It is drawing showing the condition of the in-house data of the midst which stores XML data in RDB.
[Drawing 11] It is drawing showing the contents of RDB after storing XML data.

[Drawing 12] It is drawing showing the condition of the in-house data of the midst which has taken out XML data from RDB.
[Drawing 13] It is drawing showing the condition of the in-house data of the midst which has taken out XML data from RDB.
[Description of Notations]
100 Enclosure to DB of XML Data
101 Storing Means to DB of XML Data
102 In-house Data
103 XML Data of Input
104 Input Unit
106 RDB
110 Record Medium
200 XML Data Acquisition Equipment from DB
201 XML Data Acquisition Means from DB
202 In-house Data
203 Acquired XML Data
204 Output Unit
205 RDB
210 Record Medium

[Translation done.]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号
特開2003－248615
（P2003－248615A）
(43)公開日　平成15年９月５日(2003.9.5)

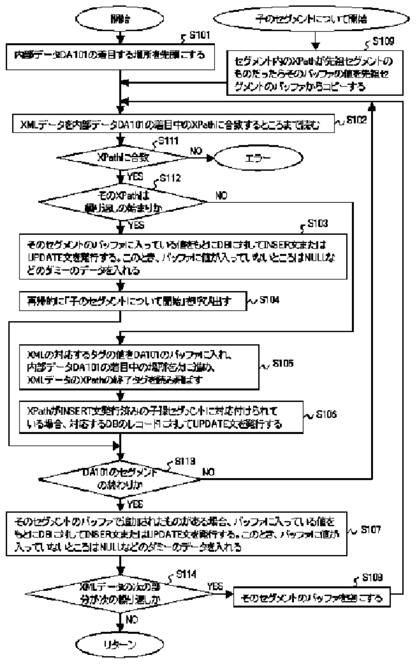
(51)Int.Cl.^{7}	識別記号	ＦＩ	テーマコード*（参考）
Ｇ０６Ｆ　12/00	５４６	Ｇ０６Ｆ　12/00	５４６Ｚ　５Ｂ０８２
	５４７		５４７Ｚ

審査請求　未請求　請求項の数８　ＯＬ（全 11 頁）

(21)出願番号	特願2002－47795(P2002－47795)
(22)出願日	平成14年２月25日(2002.2.25)
(71)出願人	000004237 日本電気株式会社 東京都港区芝五丁目７番１号
(72)発明者	稲本　直太 東京都港区芝五丁目７番１号　日本電気株式会社内
(74)代理人	100088959 弁理士　境　廣巳
Ｆターム(参考)	5B082 GA07 GA08 GA16

(54)【発明の名称】　ＸＭＬデータのＲＤＢへの格納装置及び方法、ＲＤＢからのＸＭＬデータの取得装置及び方法並びにプログラム

(57)【要約】
【課題】　繰り返し項目が多いXMLデータをRDBに格納するのに必要なバッファ量を削減する。
【解決手段】　格納対象となるXMLデータの構造を表しRDBとの対応関係を持ちXMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを順番にたどると同時にXMLデータを読みながらRDBに対してデータ追加操作を行うSQL文を発行することにより、XMLデータをRDBへ格納する。その際、XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点でRDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコードを挿入する（ステップS107）。ダミーの値を書き出したカラムの値に対応するタグの値をXMLデータから読み出したとき、UPDATE文を発行して前記レコードを更新する（ステップS106）。

【図２】

開始
子のセグメントについて開始
S101
内部データDA101の着目する場所を先頭にする
S109
セグメント内のXPathが先祖セグメントのものだったらそのバッファの値を先祖セグメントのバッファからコピーする
XMLデータを内部データDA101の着目中のXPathに合致するところまで読む
S102
S111
XPathに合致
NO
エラー
YES
S112
そのXPathは繰り返しの始まりか
NO
YES
S103
そのセグメントのバッファに入っている値をもとにDBに対してINSERT文またはUPDATE文を発行する。このとき、バッファに値が入っていないところはNULLなどのダミーのデータを入れる
再帰的に「子のセグメントについて開始」を呼び出す
S104
XMLの対応するタグの値をDA101のバッファに入れ、内部データDA101の着目中の場所を次に進め、XMLデータのXPathの終了タグを読み飛ばす
S105
XPathがINSERT文発行済みの子孫セグメントに対応付けられている場合、対応するDBのレコードに対してUPDATE文を発行する
S106
S113
DA101のセグメントの終わりか
NO
YES
そのセグメントのバッファで追加されたものがある場合、バッファに入っている値をもとにDBに対してINSERT文またはUPDATE文を発行する。このとき、バッファに値が入っていないところはNULLなどのダミーのデータを入れる
S107
S114
XMLデータの次の部分が次の繰り返しか
YES
S108
そのセグメントのバッファを空にする
NO
リターン

【特許請求の範囲】
【請求項１】　格納対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを記憶するメモリと、前記内部データを順番にたどると同時に前記XMLデータを読みながら前記RDBに対してデータ追加操作を行うSQL文を発行することにより前記XMLデータを前記RDBへ格納する格納手段とを備えたXMLデータのRDBへの格納装置。
【請求項２】　前記格納手段は、前記XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点で前記RDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコードを挿入する手段と、前記ダミーの値を書き出したカラムの値に対応するタグの値を前記XMLデータから読み出したときにUPDATE文を発行して前記レコードを更新する手段とを含むことを特徴とするXMLデータのRDBへの格納装置。
【請求項３】　取得対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを記憶するメモリと、前記内部データを順番にたどると同時に前記XMLデータを書き出しながら前記RDBに対してデータ取得操作を行うSQL文を発行することにより前記RDBから前記XMLデータを取得する取得手段とを備えたRDBからのXMLデータの取得装置。
【請求項４】　前記取得手段は、前記XMLデータの繰り返し部分については、当該繰り返し部分のタグの値を格納するテーブルに対してSELECT文を発行し、１レコード毎に、当該レコード中のカラムの値を該当するバッファに蓄積した後、当該繰り返し部分のタグに対応する値を前記バッファから読み出して前記XMLデータのストリームに書き出す処理を繰り返す手段を含む請求項１に記載のRDBからのXMLデータの取得装置。
【請求項５】　格納対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを順番にたどると同時に前記XMLデータを読みながら前記RDBに対してデータ追加操作を行うSQL文を発行することにより前記XMLデータを前記RDBへ格納する格納ステップを有し、且つ、前記格納ステップは、前記XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点で前記RDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコードを挿入するステップと、前記ダミーの値を書き出したカラムの値に対応するタグの値を前記XMLデータから読み出したときにUPDATE文を発行して前記レコードを更新するステップとを含むことを特徴とするXMLデータのRDBへの格納方法。

【請求項６】　取得対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを順番にたどると同時に前記XMLデータを書き出しながら前記RDBに対してデータ取得操作を行うSQL文を発行することにより前記RDBから前記XMLデータを取得する取得ステップを有し、且つ、前記取得ステップは、前記XMLデータの繰り返し部分については、当該繰り返し部分のタグの値を格納するテーブルに対してSELECT文を発行し、１レコード毎に、当該レコード中のカラムの値を該当するバッファに蓄積した後、当該繰り返し部分のタグに対応する値を前記バッファから読み出して前記XMLデータのストリームに書き出す処理を繰り返すステップを含むRDBからのXMLデータの取得方法。
【請求項７】　XMLデータのRDBへの格納装置を構成するコンピュータを、格納対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを順番にたどると同時に前記XMLデータを読みながら前記RDBに対してデータ追加操作を行うSQL文を発行することにより前記XMLデータを前記RDBへ格納する格納手段であって、前記XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点で前記RDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコードを挿入する手段と、前記ダミーの値を書き出したカラムの値に対応するタグの値を前記XMLデータから読み出したときにUPDATE文を発行して前記レコードを更新する手段とを含む格納手段として機能させるプログラム。
【請求項８】　RDBからのXMLデータの取得装置を構成するコンピュータを、取得対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを順番にたどると同時に前記XMLデータを書き出しながら前記RDBに対してデータ取得操作を行うSQL文を発行することにより前記RDBから前記XMLデータを取得する取得手段であって、前記XMLデータの繰り返し部分については、当該繰り返し部分のタグの値を格納するテーブルに対してSELECT文を発行し、１レコード毎に、当該レコード中のカラムの値を該当するバッファに蓄積した後、当該繰り返し部分のタグに対応する値を前記バッファから読み出して前記XMLデータのストリームに書き出す処理を繰り返す手段を含む取得手段として機能させるプログラム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明はXMLデータをRDBに格

納する装置およびXMLデータをRDBから取得する装置に関する。

【０００２】

【従来の技術】XML（eXtensible Markup Language）で記述されたデータをRDB（Relational Data Base）に格納する際、XMLデータをそのまま全部ストリーム（Stream）で格納するのではなく、検索などが有効に行えるように、RDBの各カラムとXMLの各タグの値とを対応付けて幾つかのテーブルに格納する。このため、従来のこの種の格納装置は、RDBに格納しやすい他の形式のXMLデータにXMLデータを一括して変換してバッファに蓄積してからRDBに格納している。また、こうして格納されたXMLデータをRDBから取得する際には、XMLデータを一括して取り出してバッファに蓄積し、格納しやすい形式から目的のXMLデータの形式に変換している。RDBに格納しやすい形式のXMLデータは、通常XMLデータのタグがRDBのカラム名、タグの値がRDBのカラムの内容となる以下のような形式のXMLデータである。

【０００３】<Data>
<Record1>
<Column1> value1-1 </Column1>
<Column2> value1-2 </Column2>
<Column3> value1-3 </Column3>
</Record1>
<Record2>
<Column1> value2-1 </Column1>
<Column2> value2-2 </Column2>
<Column3> value2-3 </Column3>
</Record2>
</Data>

【０００４】

【発明が解決しようとする課題】このような構成を有する従来の装置では、通常XSLT（eXtensible Style Language Transformation）プロセッサやパーズツリーであるDOM（Document Object Model）オブジェクトを使用するが、一般的にXSLTプロセッサまたはDOMオブジェクトがバッファとして使用するメモリ容量は、扱うXMLデータが大きくなればそれにつれて大きくなり、装置全体が使用するメモリ量をXMLデータのサイズによらずXMLの構造を表すデータ（DTD(Document Type Definition)またはXMLShema)の大きさと同じオーダーの一定値に保つことができない。

【０００５】その理由は、XMLデータを他形式のXMLデータに一括して変換してバッファに蓄積した後にDBに格納するため、繰り返し項目が多数あるサイズの大きいXMLデータについてこれを行う場合、使用メモリ量がXMLデータのサイズにほとんどの場合比例するからである。

【０００６】

【発明の目的】本発明の目的は、繰り返し項目が多いXMLデータをRDBに格納する場合に、XMLデータに繰り返し

項目が多数あっても使用メモリ量をXMLデータのサイズによらず、XMLの構造を表すデータDTD等の大きさと同じオーダーの一定値に保つことができるようにすることにある。

【０００７】本発明の別の目的は、繰り返し項目が多いXMLデータをRDBから取得する場合に、XMLデータに繰り返し項目が多数あっても使用メモリ量をXMLデータのサイズによらず、XMLの構造を表すデータDTD等の大きさと同じオーダーの一定値に保つことができるようにすることにある。

【０００８】

【課題を解決するための手段】本発明のXMLデータのRDBへの格納装置は、格納対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを記憶するメモリと、前記内部データを順番にたどると同時に前記XMLデータを読みながら前記RDBに対してデータ追加操作を行うSQL文を発行することにより前記XMLデータを前記RDBへ格納する格納手段とを備えている。より具体的には、前記格納手段は、前記XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点で前記RDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコードを挿入する手段と、前記ダミーの値を書き出したカラムの値に対応するタグの値を前記XMLデータから読み出したときにUPDATE文を発行して前記レコードを更新する手段とを含んでいる。

【０００９】また本発明のRDBからのXMLデータの取得装置は、取得対象となるXMLデータの構造を表しRDBとの対応関係を持ち前記XMLデータの繰り返し部分の数に関し一組分だけのバッファを持つ内部データを記憶するメモリと、前記内部データを順番にたどると同時に前記XMLデータを書き出しながら前記RDBに対してデータ取得操作を行うSQL文を発行することにより前記RDBから前記XMLデータを取得する取得手段とを備えている。より具体的には、前記取得手段は、前記XMLデータの繰り返し部分については、当該繰り返し部分のタグの値を格納するテーブルに対してSELECT文を発行し、１レコード毎に、当該レコード中のカラムの値を該当するバッファに蓄積した後、当該繰り返し部分のタグに対応する値を前記バッファから読み出して前記XMLデータのストリームに書き出す処理を繰り返す手段を含んでいる。

【００１０】

【作用】本発明のXMLデータのRDBからの取得装置にあっては、XMLデータの繰り返し部分については、１つの繰り返し部分毎に、タグの値をバッファに蓄積した時点でRDBの当該繰り返し部分のタグの値を格納するテーブルに対して、レコード中の値が未だ定まっていないカラムにはダミーの値を使用したINSERT文を発行してレコード

を挿入する。そして、ダミーの値を書き出したカラムの値に対応するタグの値をXMLデータから読み出したときにUPDATE文を発行して更新する。これにより、XMLデータの繰り返し部分のタグの値を格納するRDBのテーブルが、その繰り返し部分に含まれない外のタグの値をも格納する場合、当該テーブルに対するデータの書き出しがINSERT文、UPDATE文による複数回に別れるものの、XMLデータの繰り返し部分の数に関し一組分だけのバッファだけで足り、XMLデータのサイズがいくら大きくなっても内部データのサイズはDTDの大きさを超えることがない。

【００１１】また本発明のRDBからのXMLデータの取得装置にあっては、XMLデータの繰り返し部分については、当該繰り返し部分のタグの値を格納するテーブルに対してSELECT文を発行し、１レコード毎に、当該レコード中のカラムの値を該当するバッファに蓄積した後、当該繰り返し部分のタグに対応する値を前記バッファから読み出して前記XMLデータのストリームに書き出す処理を繰り返すため、XMLデータの繰り返し部分の数に関し一組分だけのバッファだけで足り、XMLデータのサイズがいくら大きくなっても内部データのサイズはDTDの大きさを超えることがない。

【００１２】

【発明の実施の形態】次に、本発明の実施の形態について図面を参照して詳細に説明する。

【００１３】図１を参照すると、本発明の実施の形態におけるXMLデータのRDBへの格納装置１００は、XMLデータのRDBへの格納手段１０１と、内部データDA１０１を記憶するメモリ１０２とを備え、格納対象となるXMLデータ１０３を記憶する入力装置１０４とXMLデータを格納するRDB１０５とに接続されている。入力装置１０４は例えば磁気ディスク、通信バッファ等である。なお、１１０は磁気ディスク、半導体メモリ等のコンピュータ可読記録媒体であり、格納用プログラムが記録されている。ここに記録された格納用プログラムは、格納装置１００を構成するコンピュータに読み取られ、そのコンピュータの動作を制御することにより、そのコンピュータ上にモジュールとしての格納手段１０１を実現する。

【００１４】内部データDA１０１は、入力対象となるXMLデータ１０３毎に事前に用意されており、XMLデータ１０３の構造を表し、RDB１０５との対応関係を持ち、XMLデータ１０３の中の繰り返しに関し一組分だけのバッファを持っている。より具体的には、内部データDA１０１は、XMLデータ１０３の構造を表すデータ（DTDまたはXMLShema）について、XMLデータ１０３の繰り返し部分を子セグメントにした木構造の構造になっており、値が入りうるXMLのタグについて、そのXPathと、それと対応付けられているRDB１０５のテーブル名及びカラム名と、値を保持するバッファとを持っている。親セグメントと子セグメントとはポインタで連結される。

【００１５】格納手段１０１は、内部データDA１０１を順番にたどると同時に入力のXMLデータ１０３を読みながらRDB１０５に対してデータ追加操作を行うSQL文を発行するように動作する。このときにXMLデータ１０３のサイズがいくら大きくなっても内部データDA１０１のサイズは、DTDの大きさを超えることがない。

【００１６】また、本発明の実施の形態におけるRDBからのXMLデータの取得装置２００は、RDBからのXMLデータの取得手段２０１と、内部データDA２０１を記憶するメモリ２０２とを備え、XMLデータを格納するRDB２０５と取得したXMLデータ２０３を記憶する出力装置２０４とに接続されている。出力装置２０４は例えば磁気ディスク、通信バッファ等である。なお、２１０は磁気ディスク、半導体メモリ等のコンピュータ可読記録媒体であり、取得用プログラムが記録されている。ここに記録された取得用プログラムは、取得装置２００を構成するコンピュータに読み取られ、そのコンピュータの動作を制御することにより、そのコンピュータ上にモジュールとしての取得手段２０１を実現する。

【００１７】内部データDA２０１は、取得対象となるXMLデータ２０３毎に事前に用意されており、XMLデータ２０３の構造を表し、RDB２０５との対応関係を持ち、XMLデータ２０３の中の繰り返しに関し一組分だけのバッファを持っている。より具体的には、内部データDA２０１は、XMLデータ２０３の構造を表すデータ（DTDまたはXMLShema）について、XMLデータ２０３の繰り返し部分を子セグメントにした木構造の構造になっており、値が入りうるXMLのタグについて、そのXPathと、それと対応付けられているRDB２０５のテーブル名及びカラム名と、値を保持するバッファとを持っている。親セグメントと子セグメントとはポインタで連結される。

【００１８】取得手段２０１は、内部データDA２０１を順番にたどると同時にXMLデータ２０３を書き出しながらRDB２０５に対してデータ取得操作を行うSQL文を発行するよう動作する。このときにXMLデータ２０３のサイズがいくら大きくなっても内部データDA２０１のサイズは、DTDの大きさを超えることがない。

【００１９】次に、図２のフローチャートを参照して本実施の形態の格納装置１００の全体の動作について詳細に説明する。

【００２０】まず、内部データDA１０１の着目する場所を先頭にする（図２のステップS１０１）。内部データDA１０１はセグメントの木構造になっているため、最初はルートのセグメントの先頭を着目する場所に定める。次に、XMLデータ１０３を内部データDA１０１の着目中のXPathに合致するところまで読む（ステップS１０２）。ここでXPathに合致するところがXMLデータ１０３になかったら（ステップS１１１でＮＯ）、エラーにする。さらに、そのXPathに対応するタグが繰り返しの始まりかチェックし（ステップS１１２）、繰り返しの始

まりの場合はステップS１０３に進み、繰り返しの始まりでない場合はステップS１０５に進む。繰り返しの始まりの箇所には子セグメントへのポインタが入っているので、ポインタの有無により繰り返しの始まりか否かが判明する。

【００２１】ステップS１０３では、親セグメントについてバッファに入っている値を元に、RDB１０５の該当するテーブルに対してINSERT文またはUPDATE文を発行する。INSERT文は当該親セグメントにおける初めての子セグメントの開始時（繰り返しの始まり）に使用され、UPDATE文は当該親セグメントにおける２番目以降の子セグメントの開始時（繰り返しの始まり）に使用される。INSERT文では、テーブルに１行が挿入されるが、その行の全てのカラムの値がバッファに入っていなければ、値が入っていないところはNULLなどのダミーのデータを入れる。次に、内部データDA１０１のそれに対応する子セグメントについて、「子セグメントについて開始」から始まる部分を再帰的に実行する（ステップS１０４）。そして、ステップS１１３に進む。

【００２２】他方、ステップS１０５では、XMLデータ１０３からXPathに対応する値を読んで内部データDA１０１のバッファに格納し、XMLデータ１０３のXPathに対応するXMLの終了タグを読み飛ばし、内部データDA１０１の着目する場所を次に進める。次に、もしも着目していたXPathが、RDB１０５のテーブルのカラムが子孫のセグメントからINSERT文を発行したDBレコードに対応付けられていた場合は、それらのDBレコードに対してUPDATE文を発行する（ステップS１０６）。そして、ステップS１１３に進む。

【００２３】ステップS１１３では、内部データの着目場所が着目中のセグメントの終わりかどうかチェックし、終わりでない場合はステップS１０２からの処理を繰り返し実行する。終わりの場合には、当該セグメントのバッファに追加された値がある場合は、バッファに入っている値を元にRDB１０５に対して、初めての書き出しの際にはINSERT文を発行し、２度目以降の同じレコードに対する書き出しの際にはUPDATE文を発行する（ステップS１０７）。そして、XMLデータ１０３の次の部分がそのセグメントを繰り返すタグになっているかチェックし、繰り返しになっている場合は、そのセグメントのバッファを再利用するためにステップS１０９でコピーしたバッファ以外のバッファを空にし（ステップS１０８）、ステップS１０２からの処理を繰り返し実行する。繰り返しでない場合はリターンする。なお、ステップS１０８でそのセグメントのバッファを全て空にするようにしても良いが、その場合にはステップS１０９からの処理を繰り返す。

【００２４】このように本実施の形態の格納装置１００では、XMLデータ１０３をRDB１０５に格納する場合に、中間的にメモリ１０２上に持つ値はXMLデータ１０３の構造（DTD）の大きさとなるように構成されているため、たとえ繰り返し項目が多数あってXMLデータ１０３のサイズがいくら大きくなっても、中間的にメモリ１０２上に持つデータのサイズを入力XMLデータ１０３のサイズによらずDTDの大きさと同じオーダーの一定値にできる。

【００２５】次に、図３のフローチャートを参照して本実施の形態の取得装置２００の全体の動作について詳細に説明する。

【００２６】まず、内部データDA２０１の着目する場所を先頭にする（図３のステップS２０１）。内部データDA２０１はセグメントの木構造になっているため、最初はルートのセグメントの先頭を着目する場所に定める。次に、そのセグメントの着目中のXPathに対応付けられているRDB２０５のテーブルに対しSELECT文を発行する（ステップS２０２）。次に、ステップS２０２で発行したSQL文の実行結果の次のレコードがあるかチェックする（ステップS２０３）。次のレコードがない場合（ステップS２１１でNO）、リターンする。次のレコードがある場合は、ステップS２０２で発行したSQL文の実行結果の次のレコードをフェッチし、得られた値を内部データDA２０１のそのセグメントのバッファに入れる（ステップS２０４）。

【００２７】次に、内部データDA２０１の着目場所のXPathがXMLデータ２０３の繰り返しに対応するタグかどうかチェックし（ステップS２１２）、繰り返しの始まりの場合はステップS２０５へ進み、繰り返しの始まりでない場合はステップS２０６に進む。繰り返しの始まりの箇所には子セグメントへのポインタが入っているので、ポインタの有無により繰り返しの始まりか否かが判明する。

【００２８】ステップS２０５では、そのXPathに対応する内部データDA２０１の子セグメントについて、「子セグメントについて開始」からの処理を再帰的に実行する。そして、ステップS２１３に進む。

【００２９】ステップS２０６では、内部データDA２０１の着目中のXPathに対応するバッファの値が空の場合にはRDB２０５に対してSELECT文を発行してバッファの値を埋める。そして、XPathに対応するタグとバッファの値をXMLデータ２０３に書き出し、内部データDA２０１の着目中の場所を次へ進める（ステップS２０７）。そして、ステップS２１３に進む。

【００３０】ステップS２１３では、内部データDA２０１の着目中の場所がそのセグメントの終わりかどうかチェックする。セグメントの終わりでない場合はステップS２１２からの処理を繰り返し実行する。セグメントの終わりの場合は、ステップS２０３からの処理を繰り返し実行する。

【００３１】本実施の形態の取得装置２００では、XMLデータ２０３をRDB２０５から取得する場合、XMLデータ

２０３はストリームとして書き出してしまい、中間的にメモリ２０２上に持つ値はXMLデータ２０３の構造（DTD）の大きさとなるように構成されているため、たとえ繰り返し項目が多数あってXMLデータ２０３のサイズがいくら大きくなっても、中間的にメモリ２０２上に持つデータのサイズをXMLデータ２０３のサイズによらずDTDの大きさと同じオーダーの一定値にできる。

【００３２】次に、具体的な実施例を用いて本実施の形態の動作を説明する。

【００３３】図４はXMLデータのDTDと、RDBのDBスキーマの具体例を示す。DTDでは要素型宣言部分のみを示してあり、要素Itemが繰り返される構造となっている。また、RDBは、PORequestテーブルとPOItemテーブルとの２つのテーブルを持つ。

【００３４】図５はXML中のタグとDBテーブルのカラムとの対応関係の具体例を示す。この例では、値が入りうるXMLのタグのうち、Xpathが“/POReq/PONumber”，“/POReq/PODate”，“/POReq/fromRole/BusinessId”，“/POReq/toRole/BusinessId”であるタグの値が、PORequestテーブルのカラムPOId,Date,From,Toの値に対応する。また、XMLの繰り返し部分のタグItemに含まれる、Xpathが、“/POReq/Item/POItemNumber”，“/POReq/Item/ProductId”，“/POReq/Item/Request/Amount”，“/POReq/Item/Request/Price”であるタグの値と、当該繰り返し部分の外にある、Xpathが“/POReq/Location”であるタグの値とが、POItemテーブルのカラムPOId,ItemNumber,PId,Quantity,RequestPrice,Locationの値に対応する。なお、PORequestテーブルのカラムPOIdとPOItemテーブルのカラムPOIdとは、プライマリキーとフォーリンキーの関係にある。

【００３５】図６は格納対象となるXMLデータの具体例を示す。この例では繰り返し項目１～３の部分が繰り返し構造となっている。

【００３６】図７は内部データの具体例を示す。この例の内部データは、図６のXMLデータの構造を表す図４のDTDについて、XMLデータの繰り返し部分を子セグメント（セグメント２）、それ以外を親セグメント（セグメント１）とした木構造を有し、各セグメント１、２において、図５の対応関係に従って、値が入りうるXMLのタグについて、そのXPathと、それと対応付けられているRDBのテーブル名及びカラム名と、値を保持するバッファとを設定してある。このような内部データは例えば以下のように作成される。

【００３７】先ず、親セグメント１を生成する。次に、図５を参照すると、XMLの値が入りうる最初のXpathは“/POReq/PONumber”で、PORequestテーブルのカラムPOIdに対応するので、エントリE1を親セグメント１に生成する。同様にして、エントリE2～E4を生成する。XMLの次のタグは繰り返しのタグItemなので、子セグメント２を生成し、親セグメント１にはタグItemのXpathと子セグメント２へのポインタを設定したエントリE5を生成する。次に、子セグメント２のエントリの生成に移るが、図５を参照すると、PORequestテーブルのカラムPOIdとPOItemテーブルのカラムPOIdとがプライマリキーとフォーリンキーの関係にあるため、プライマリキーに対応するXpathと、フォーリンキーを格納するテーブル及びカラムと、バッファとを持つエントリE50を生成した後、XMLの値が入りうる各Xpath用のエントリE51～E54を生成する。次に、親セグメント１のエントリの生成に戻り、XMLの値が入りうる最後のXpath用のエントリE6を生成する。ここで、初期状態においては各エントリのバッファは空である。

【００３８】以下、図４のDTDとDBスキーマで、これらの間に図５に示す対応関係があり、入力するXMLデータが図６に示すもので、図７のような内部データが用意されている場合を例にして、格納装置１００の具体的な動作を説明する。

【００３９】XMLデータを読み始める前の内部データDA１０１の初期状態は図７に示すものとなっており、この中のセグメント１の先頭のエントリE1が着目する場所としてセットされる（ステップS１０１）。次に、この場所に対応するXPath “/POReq/PONumber” に対応するところをXMLデータから読み出す（ステップS１０２）。先頭が対応するタグとなっているのでエラーにはならず、またこのXPathはXMLデータ中の繰り返しに対応するXPathではないのでステップS１０５に進む。

【００４０】ステップS１０５では、XMLデータの対応するタグの値 “order1” を内部データの対応するエントリE1のバッファに格納し、内部データの着目する場所を次に進める。着目中だったXPathが対応付けられているDBカラムは、子孫のセグメントでINSERT文を発行したレコードのものではないので、これに対するUPDATE文は発行されない（ステップS１０６）。次に、セグメントの終わりではないので、ステップS１０２からの処理を繰り返し実行する。XPath “/POReq/PODate”，“/POReq/fromRole/BusinessId”，“/POReq/toRole/BusinessId” について同様のこと（ステップS１０２、ステップS１０５）を繰り返した結果、内部データは図８に示すものとなる。

【００４１】次のXPath “/POReq/Item” については、このXPathがXMLデータ中の繰り返しに対応するので、子セグメント２に基づく処理（繰り返し部分の処理）に入る前の親セグメントに対する処理として、ステップS１０３の処理を実行する。具体的には、現在の内部データのセグメント１のバッファに入っている値を元にDBのPORequestテーブルに対しINSERT文を発行する（ステップS１０３）。ここで、操作対象となるテーブルはセグメント１のバッファが空でないものに対応するテーブルだけである。このため、PORequestテーブルだけが対象となり、エントリE6のバッファが空であるPOItemテーブルに

対する操作は行われない。

【００４２】次に、このXPath “/POReq/Item” に対応する子セグメント２についてステップS１０２からの処理が再帰的に実行される（ステップS１０４）。

【００４３】先ず、子セグメント２の最初の XPath ”/POReq/PONumber” は、”/POReq/Item” 以下のパスではなく先祖セグメントのXPathに対応したものであり、DBテーブルではプライマリキーとフォーリンキーの関係にあるものとなっており（これは、子セグメント中のXpathと同じXpathが親セグメントにあるかどうかで判明する）、この場合は、バッファに親のセグメントのバッファに入っている値 “order1” をコピーし（ステップS１０９）、次のXPath “POItemNumber” に進む。すなわち、親セグメントのバッファの値が、DBテーブルの親テーブルと子テーブルを関係つけるためのプライマリキーであれば、子セグメントに子テーブルの対応するフォーリンキーの項目があるため、ここに値をコピーするわけである。

【００４４】次に、子セグメント２におけるXPath “POItemNumber”, “ProductId”, “Request/Amout”, “Request/Price” についてステップS１０２、ステップS１０５が繰り返し実行された結果、内部データは図９に示すものとなる。

【００４５】これでセグメント２の終わりなので、セグメントの終了時の処理を行う（ステップS１０７）。今の場合、セグメント２のバッファに入っている値を元にDBのPOItemテーブルに対してINSERT文が発行される。このとき、テーブルPOItemテーブルのカラムLocationに対応する値は、エントリE6のバッファに未だ入っていないので、このカラムについてはNULLなどのダミーのデータが使用される。

【００４６】図６に示すXMLデータは、繰り返しのXPathの “/POReq/Item” が３つ含まれているので、同様のこと（ステップS１０２、ステップS１０５、ステップS１０７）をあと２回繰り返して、POItemテーブルに対してあと２回INSERT文が発行され、子セグメント２についての処理からセグメント１への処理へリターンする。

【００４７】セグメント１の次のXPathは “/POReq/Location” で、これに関しステップS１０２とステップS１０５を実行した結果、内部データは図１０に示すものとなる。そして、”/POReq/Location” は、セグメント２に対応するDBのレコードのカラムに対応付けられているので、既にINSERT文が発行済みのPOItemテーブルのレコードに対して、このバッファに入っている値を元に、DBのPOItemテーブルに対してUPDATE文が発行される（ステップS１０６）。次にステップS１０７の実行に入るが、ステップS１０７は「そのセグメントのバッファで追加されたものがある場合」だけ処理を行うものであり、今の場合はバッファに新たに追加されたものがないため、何もしない。これでセグメント１の終わりで、XMLデータのRDBへの格納処理を終了する。このときにRDBに格納されているデータは図１１に示すものとなる。

【００４８】以上の具体例では、ステップS１０３においてNULLなどのダミーのデータを入れる場面、同ステップS１０３においてUPDATE文を使用する場面、ステップS１０７においてUPDATE文を使用する場面は現れない。これらの場面は、例えば次のような具体例で現れる。

【００４９】例えば図７のエントリE2がエントリE5とE6の間にあるような場合を考える。このような場合、親セグメント１のエントリE4までの処理を終えて、繰り返し部分の処理に入る前の親セグメント１に対するステップS１０３において、エントリE2のバッファが未だ空なので、PORequestテーブルのカラムDateにはNULLなどのダミーのデータが入れられる。また、子セグメント１の処理を終えて、エントリE2の処理に移ると、ステップS１０５でエントリE2のバッファに値が入るため、親セグメント１の処理を終える際のステップS１０７において、エントリE2のバッファに入っている値がUPDATE文を使用して、PORequestテーブルのカラムDateに書き込まれる。

【００５０】また、ステップS１０３でUPDATE文が使用されるのは、親セグメント１に複数の繰り返し部分がある場合である。すなわち、１個目の子セグメントの処理に入る際の親セグメントに対するステップS１０３ではINSERT文が使用されるが、２個目以降の子セグメントの処理に入る際の親セグメントに対するステップS１０３の処理ではUPDATE文が使用される。

【００５１】次に、図４のDTDとDBスキーマで、これらの間に図５に示す対応関係があり、RDBに格納されているデータが図１１に示すもので、図７のような内部データが用意されている場合を例にして、取得装置２００の具体的な動作を説明する。

【００５２】XMLデータの書き出しを始める前の内部データDA２０１の初期状態は図７に示すものとなっており、この中のセグメント１の先頭のエントリE1が着目する場所としてセットされる（ステップS２０１）。次に、このセグメント１で対応付けられているDBカラムの値を取得するSELECT文が図１１で示される内容のDBのPORequestテーブルに対し実行される（ステップS２０２）。レコードが存在するので、ステップS２０４に進む（ステップS２０３）。SELECT文の実行結果をフェッチすることにより、セグメント１のエントリE1～E4のバッファの値が埋まる（ステップS２０４）。この時点で内部データDA２０１は図１２に示すものとなる。

【００５３】最初のXPath “POReq/PONumber” は繰り返しの項目ではないのでステップS２０６に進む。対応するエントリE1のバッファは空でないのでSELECT文は発行されない（ステップS２０６）。内部データDA２０１のXPath “POReq/PONumber” に対応するタグとその値がXMLデータに書き出され、次のXPathに進む（ステップS２０

７）。ここで、XMLデータに書き出すときのXMLの終了タグについては、そのタグに囲まれたところが閉じられるまで書き出しが遅延される。

【００５４】次に、XPath “/POReq/POData”，“/POReq/fromRole/BusinessId”，“/POReq/toRole/BusinessId” について同様のこと（ステップS２０７）を繰り返した結果、XMLデータの繰り返しに対応するXPath “/POReq/Item” の前までのXMLデータが書き出される。

【００５５】次のXPath “/POReq/Item” はXMLデータ中の繰り返しに対応するので、XPath “/POReq/Item” に対応する子セグメント２について「子セグメントについて開始」からの処理が再帰的に実行される（ステップS２０５）。子セグメント２のXPath”/POReq/PONumber”は、”/POReq/Item” 以下のパスではなく先祖セグメントのXPathに対応したものであり、DBテーブルではプライマリキーとフォーリンキーの関係にあるものとなっており、この場合は、エントリE50のバッファに親セグメントのエントリE1のバッファに入っている値 “order 1” がコピーされ、XPath “POItemNumber” に進む（ステップS２０８）。次に、このセグメント２で対応付けられているDBカラムの値を取得するSELECT文が図１１で示される内容のDBのPOItemテーブルに対し実行される（ステップS２０２）。レコードが存在するので、ステップS２０４に進む（ステップS２０３）。SELECT文の実行結果をフェッチすることにより、セグメント２のエントリE51～E54のバッファの値が埋まる（ステップS２０４）。

【００５６】次に、XPath “POItemNumber”，“ProductId”，“Request/Amout”，“Request/Price” についてステップS２０７が繰り返し実行された結果、XPath “POReq/Item” に対応する繰り返し項目の１つ目がXMLデータに書き出される。これでセグメント２の終わりなのでステップS２０３からの処理を繰り返す。図１１に示すDBのPOItemテーブルは３つのレコードを含んでいるので、同様のこと（ステップS２０３、ステップS２０４、ステップS２０７）をあと２回繰り返して、残りの２つの繰り返し項目に対応する内容をXMLデータに書き出し、子セグメント２についての処理からセグメント１への処理へリターンする。

【００５７】セグメント１の次のXPathは “/POReq/Location” で、これに対応するエントリE6のバッファが空なので、DBのPOItemテーブルにSELECT文を発行しフェッチしてバッファを埋め（ステップS２０６）、ステップS２０７を実行し対応する内容がXMLデータに書き出される。これで、XMLデータのRDBからの取得処理を終了する。この時点で内部データDA２０１は図１３に示すものとなり、書き出されたXMLデータは図６に示すものと同じものになる。

【００５８】以上本発明の実施の形態について説明したが、本発明は以上の例に限定されずその他各種の付加変

更が可能である。例えば、図１では、格納装置１０１と取得装置２０１とのそれぞれに内部データDA１０１、内部データDA２０１を設けたが、同じ構造のXMLデータについては格納装置１０１と取得装置２０１とで内部データは同じになるので、格納と取得とを同時に行わない場合には、１つの内部データを格納装置１０１と取得装置２０１とで共通に設けるようにしても良い。この場合、格納装置１０１と取得装置２０１とをあわせて、格納・取得装置として１つの装置として実現するようにしても良い。

【００５９】

【発明の効果】以上説明したように本発明によれば、XMLデータをRDBに格納し、またRDBからXMLデータを取得する場合に、たとえXMLデータのサイズがいくら大きくなっても、使用メモリ量がDTDのサイズに比例した大きさに抑えられ、XMLデータのサイズによらずDTDの大きさと同じオーダーの一定値に抑えられる効果がある。その理由は、既存のXML処理系ではXMLデータのサイズに比例したメモリ容量を必要とするのに対し、本発明ではDTDの大きさに比例した内部データしかメモリ上に持たず、他のデータはDBへの格納ならDBに書き出してしまうし、DBからの取り出し時はXMLデータを作成するストリームに書き出してしまうためである。

【図面の簡単な説明】

【図１】本発明の実施の形態の構成を示すブロック図である。

【図２】本発明の実施の形態のXMLデータのRDBへの格納手段の動作を示す流れ図である。

【図３】本発明の実施の形態のRDBからのXMLデータの取得手段の動作を示す流れ図である。

【図４】DTDとDBスキーマ情報の具体例を示す図である。

【図５】XML中のタグとDBテーブルのカラムとの対応関係の具体例を示す図である。

【図６】XMLデータの具体例を示す図である。

【図７】初期状態の内部データの具体例を示す図である。

【図８】RDBにXMLデータを格納している最中の内部データの状態を示す図である。

【図９】RDBにXMLデータを格納している最中の内部データの状態を示す図である。

【図１０】RDBにXMLデータを格納している最中の内部データの状態を示す図である。

【図１１】XMLデータを格納後のRDBの内容を示す図である。

【図１２】XMLデータをRDBから取り出している最中の内部データの状態を示す図である。

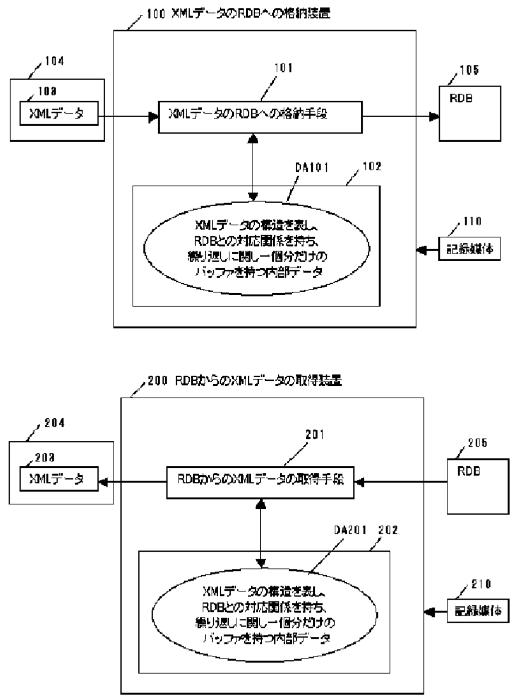
【図１３】XMLデータをRDBから取り出している最中の内部データの状態を示す図である。

【符号の説明】

１００	XMLデータのDBへの格納装置
１０１	XMLデータのDBへの格納手段
１０２	内部データ
１０３	入力のXMLデータ
１０４	入力装置
１０６	RDB
１１０	記録媒体
＊２００	DBからのXMLデータ取得装置
２０１	DBからのXMLデータ取得手段
２０２	内部データ
２０３	取得したXMLデータ
２０４	出力装置
２０５	RDB
＊　２１０	記録媒体

【図１】

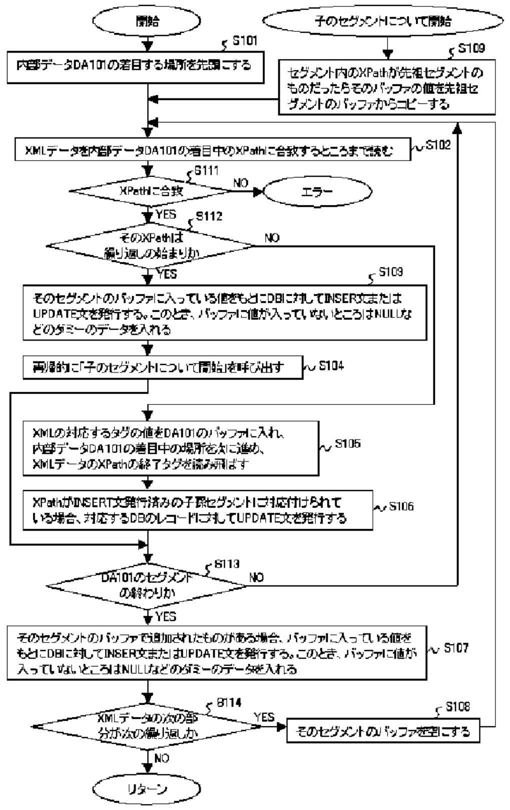
【図１】



100 XMLデータのRDBへの格納装置
104
103
XMLデータ
101
XMLデータのRDBへの格納手段
105
RDB
DA101
102
XMLデータの構造を表し、RDBとの対応関係を持ち、繰り返しに関し一個分だけのバッファを持つ内部データ
110
記録媒体
200 RDBからのXMLデータの取得装置
204
203
XMLデータ
201
RDBからのXMLデータの取得手段
205
RDB
DA201
202
XMLデータの構造を表し、RDBとの対応関係を持ち、繰り返しに関し一個分だけのバッファを持つ内部データ
210
記録媒体

【図２】

【図２】



開始
子のセグメントについて開始
S101
内部データDA101の着目する場所を先頭にする
S109
セグメント内のXPathが先祖セグメントのものだったらそのバッファの値を先祖セグメントのバッファからコピーする
XMLデータを内部データDA101の着目中のXPathに合致するところまで読む
S102
S111
XPathに合致
NO
エラー
YES
S112
そのXPathは繰り返しの始まりか
NO
YES
S103
そのセグメントのバッファに入っている値をもとにDBに対してINSER文またはUPDATE文を発行する。このとき、バッファに値が入っていないところはNULLなどのダミーのデータを入れる
再帰的に「子のセグメントについて開始」を呼び出す
S104
XMLの対応するタグの値をDA101のバッファに入れ、内部データDA101の着目中の場所を次に進め、XMLデータのXPathの終了タグを読み飛ばす
S105
XPathがINSERT文発行済みの子孫セグメントに対応付けられている場合、対応するDBのレコードに対してUPDATE文を発行する
S106
S113
DA101のセグメントの終わりか
NO
YES
そのセグメントのバッファで追加されたものがある場合、バッファに入っている値をもとにDBに対してINSER文またはUPDATE文を発行する。このとき、バッファに値が入っていないところはNULLなどのダミーのデータを入れる
S107
S114
XMLデータの次の部分が次の繰り返しか
YES
S108
そのセグメントのバッファを空にする
NO
リターン

【図４】

【図４】　DTD

```
<!ELEMENT POReq (PONumber, POData, fromRole, toRole, Item+, Location)>
<!ELEMENT fromRole (BusinessId)>
<!ELEMENT toRole (BusinessId)>
<!ELEMENT PONumber #PCDATA>
<!ELEMENT POData #PCDATA>
<!ELEMENT BusinessId #PCDATA>
<!ELEMENT Item (POItemNumber, ProductId, Request)>
<!ELEMENT Request (Amount, Price)>
<!ELEMENT POItemNumber #PCDATA>
<!ELEMENT ProductId #PCDATA>
<!ELEMENT Amount #PCDATA>
<!ELEMENT Price #PCDATA>
<!ELEMENT Location #PCDATA>
```

PORequestテーブル

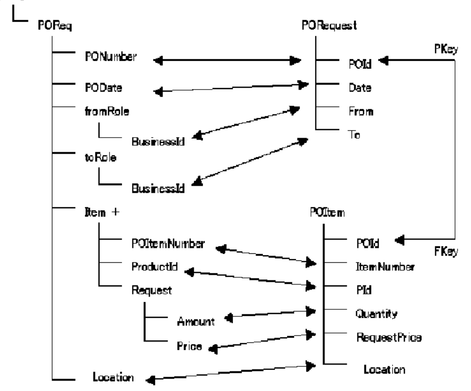
POId	Date	From	To

POItemテーブル

POId	ItemNumber	PId	Quantity	RequestPrice	Location

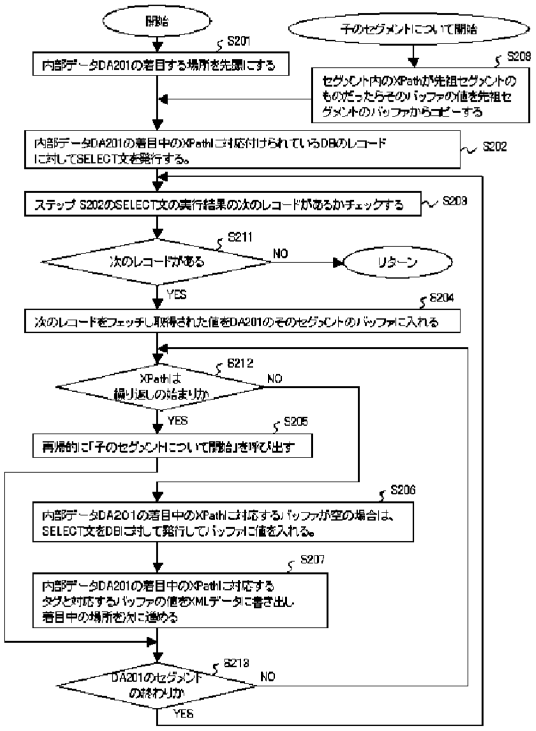
【図５】

【図５】



【図３】

【図３】


開始
子のセグメントについて開始
S201
内部データDA201の着目する場所を先頭にする
S208
セグメント内のXPathが先祖セグメントのものだったらそのバッファの値を先祖セグメントのバッファからコピーする
内部データDA201の着目中のXPathに対応付けられているDBのレコードに対してSELECT文を発行する。
S202
ステップ S202のSELECT文の実行結果の次のレコードがあるかチェックする
S203
S211
次のレコードがある
NO
リターン
YES
S204
次のレコードをフェッチし取得された値をDA201のそのセグメントのバッファに入れる
S212
XPathは繰り返しの始まりか
NO
YES
S205
再帰的に「子のセグメントについて開始」を呼び出す
S206
内部データDA201の着目中のXPathに対応するバッファが空の場合は、SELECT文をDBに対して発行してバッファに値を入れる。
S207
内部データDA201の着目中のXPathに対応するタグと対応するバッファの値をXMLデータに書き出し着目中の場所を次に進める
S213
DA201のセグメントの終わりか
NO
YES

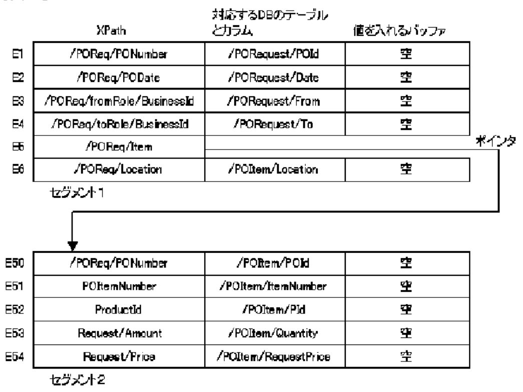
【図６】

【図６】

```
<POReq>
  <PONumber>order1</PONumber>
  <PODate>2002/7/17</PODate>
  <fromRole><BusinessId>11111111</BusinessId></fromRole>
  <toRole><BusinessId>222222222</BusinessId></toRole>
  <Item>
    <POItemNumber>Item1</POItemNumber>
    <ProductId>Desktop PC</ProductId>
    <Request>                                  繰り返し項目１
      <Amount>9</Amount>
      <Price>100000</Price>
    </Request>
  </Item>
  <Item>
    <POItemNumber>Item2</POItemNumber>
    <ProductId>Mobile Phone</ProductId>
    <Request>                                  繰り返し項目２
      <Amount>13</Amount>
      <Price>21000</Price>
    </Request>
  </Item>
  <Item>
    <POItemNumber>Item3</POItemNumber>
    <ProductId>ASDL Router</ProductId>
    <Request>                                  繰り返し項目３
      <Amount>21</Amount>
      <Price>60000</Price>
    </Request>
  </Item>
  <Location>Tokyo</Location>
</POReq>
```

【図７】

【図７】



	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	空
E2	/POReq/PODate	/PORequest/Date	空
E3	/POReq/fromRole/BusinessId	/PORequest/From	空
E4	/POReq/toRole/BusinessId	/PORequest/To	空
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	空

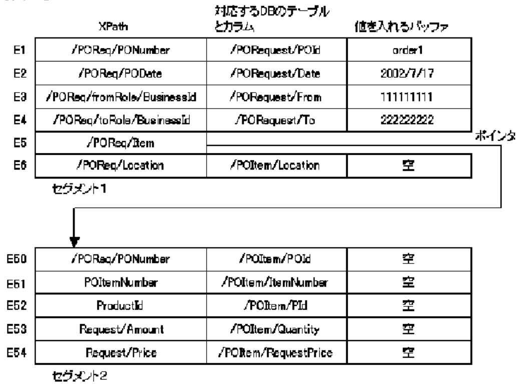
セグメント１

E50	/POReq/PONumber	/POItem/POId	空
E51	POItemNumber	/POItem/ItemNumber	空
E52	ProductId	/POItem/PId	空
E53	Request/Amount	/POItem/Quantity	空
E54	Request/Price	/POItem/RequestPrice	空

セグメント２

【図８】

【図８】



	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	order1
E2	/POReq/PODate	/PORequest/Date	2002/7/17
E3	/POReq/fromRole/BusinessId	/PORequest/From	111111111
E4	/POReq/toRole/BusinessId	/PORequest/To	222222222
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	空

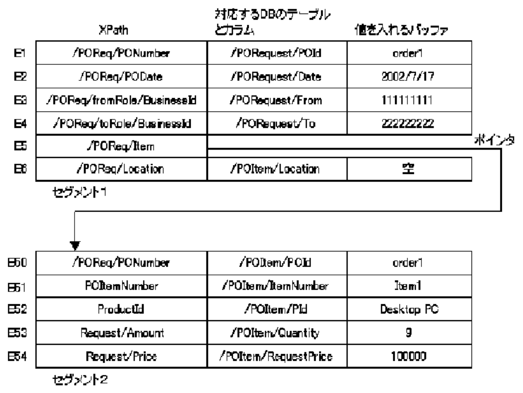
セグメント１

E50	/POReq/PONumber	/POItem/POId	空
E51	POItemNumber	/POItem/ItemNumber	空
E52	ProductId	/POItem/PId	空
E53	Request/Amount	/POItem/Quantity	空
E54	Request/Price	/POItem/RequestPrice	空

セグメント２

【図９】

[図９]

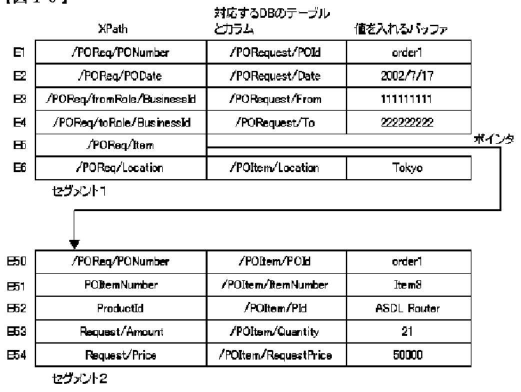


	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	order1
E2	/POReq/PODate	/PORequest/Date	2002/7/17
E3	/POReq/fromRole/BusinessId	/PORequest/From	111111111
E4	/POReq/toRole/BusinessId	/PORequest/To	222222222
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	空

E50	/POReq/PONumber	/POItem/POId	order1
E51	POItemNumber	/POItem/ItemNumber	Item1
E52	ProductId	/POItem/PId	Desktop PC
E53	Request/Amount	/POItem/Quantity	9
E54	Request/Price	/POItem/RequestPrice	100000

【図１０】

[図１０]

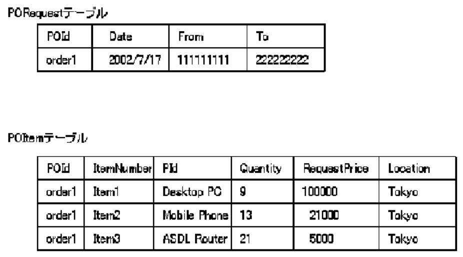


	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	order1
E2	/POReq/PODate	/PORequest/Date	2002/7/17
E3	/POReq/fromRole/BusinessId	/PORequest/From	111111111
E4	/POReq/toRole/BusinessId	/PORequest/To	222222222
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	Tokyo

E50	/POReq/PONumber	/POItem/POId	order1
E51	POItemNumber	/POItem/ItemNumber	Item8
E52	ProductId	/POItem/PId	ASDL Router
E53	Request/Amount	/POItem/Quantity	21
E54	Request/Price	/POItem/RequestPrice	50000

【図１１】

[図１１]



PORequestテーブル

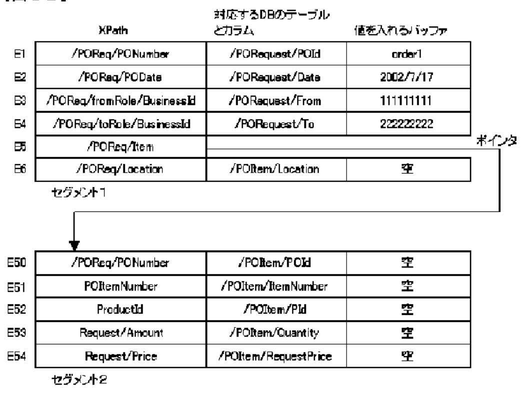
POId	Date	From	To
order1	2002/7/17	111111111	222222222

POItemテーブル

POId	ItemNumber	PId	Quantity	RequestPrice	Location
order1	Item1	Desktop PC	9	100000	Tokyo
order1	Item2	Mobile Phone	13	21000	Tokyo
order1	Item3	ASDL Router	21	5000	Tokyo

【図１２】

[図１２]

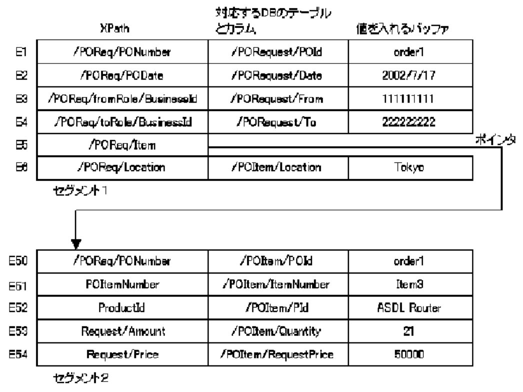


	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	order1
E2	/POReq/PODate	/PORequest/Date	2002/7/17
E3	/POReq/fromRole/BusinessId	/PORequest/From	111111111
E4	/POReq/toRole/BusinessId	/PORequest/To	222222222
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	空

E50	/POReq/PONumber	/POItem/POId	空
E51	POItemNumber	/POItem/ItemNumber	空
E52	ProductId	/POItem/PId	空
E53	Request/Amount	/POItem/Quantity	空
E54	Request/Price	/POItem/RequestPrice	空

【図１３】

[図１３]



	XPath	対応するDBのテーブルとカラム	値を入れるバッファ
E1	/POReq/PONumber	/PORequest/POId	order1
E2	/POReq/PODate	/PORequest/Date	2002/7/17
E3	/POReq/fromRole/BusinessId	/PORequest/From	111111111
E4	/POReq/toRole/BusinessId	/PORequest/To	222222222
E5	/POReq/Item		
E6	/POReq/Location	/POItem/Location	Tokyo

E50	/POReq/PONumber	/POItem/POId	order1
E51	POItemNumber	/POItem/ItemNumber	Item3
E52	ProductId	/POItem/PId	ASDL Router
E53	Request/Amount	/POItem/Quantity	21
E54	Request/Price	/POItem/RequestPrice	50000